



# **FINAL YEAR PROJECT REPORT**

## **ELECTRONIC ELECTION SYSTEM OF PAKISTAN**

Submitted By

KAMRAN UL HAQUE

JUNAID AHMED

Bachelors Of Computer Science and Information Technology

**DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION  
TECHNOLOGY**

**NED UNIVERSITY OF ENGINEERING AND TECHNOLOGY, KARACHI**



## **CERTIFICATE**

This is to certify that the following students of Batch 2002-2003 have successfully completed the final year project to fulfill the requirements for a Bachelor's Degree in Computer Science and Information Technology from NED University of Engineering and Technology, Karachi, Pakistan.

**KAMRAN UL HAQUE**

**CT-066**

**JUNAID AHMED**

**CT-072**

## **PROJECT SUPERVISORS**

---

Prof. Azam Mughal  
(Internal)  
Department of Computer Science and I.T

---

Prof. Dr. Mahmood Khan Pathan  
(Chairman)  
Department of Computer Science and IT

.....  
Syed Akbar Zaidi  
(Project External)

## ABSTRACT

The present election system of Pakistan is based on manual procedures. Votes are polled manually. All the record of voters, candidates, polling stations, regions detail and parties information are stored manually. Results are calculated manually. The process is quite primitive and has certain flaws in it. These include: Corruption, lack of accuracy, duplicate voting, unsatisfactory counting procedures, high usage of resources like money, human resource and time. The voting process is very slow. The voter has to go to various desks for different procedures. This makes the process time inefficient. For this reason, all the voters usually don't get the chance to vote in specific time period.

The best way to remove the flaws in the present election system is to devise a computerized election system via fingerprint recognition. This will improve the system's efficiency and effectiveness both. The system will make the voter independent of polling station, means he/she can vote from anywhere in the country through electronic polling stations. This will reduce the chances of error occurrence, while counting of votes, minimize time being consumed during counting, increase security of votes being polled via biometric devices. No chances of vote's rejection and easy and timely management of records. Pakistan is in need of such a system where people can vote without any problem and feel secure and no one can manipulate the results. Such environment can only be created through some automated system which cannot be cheated.

This report documents the development of the e-Election System of Pakistan – an automated public voting system to be deployed in Pakistan for Election. The project is motivated by the need to increase efficiency and removes flaws from existing election system. The system makes the voter to caste vote through electronic polling stations.

The proposed system is designed in distributed environment thus contain multiple servers in different cities within Pakistan. Each Polling station is connected via WAN/LAN to its city server. We have divided the city into several constituencies. On every constituency a client database is maintained which contain information of all voters who belong to particular constituency in horizontal partitioning with continuous replication. The application in distributed database environment also facilitates the person who is not presently living at his particular constituency, the person can cast vote to his near polling



station and the data of that person will be updated in to that particular constituency from where he/she really belongs. e.g. if the person is presently living in Islamabad and his CNIC is of Karachi he/she is allowed caste vote from Islamabad and the data will be updated at Karachi in his particular constituency where he/she belongs.

For handling security that a person may cast dual vote or may have two NICs. We use Fingerprint Recognition System via Fingerprint Scanners available at each polling station within city so that the voter will be traced or knock out by his fingerprint impressions so that he/she cannot cast dual vote.

The administration of the system is under control of Election Commission of Pakistan which is responsible to declare election results by generating the reports from the system and displaying it on their website.

---



## **1. INTRODUCTION**

As the world watched the electoral drama unfold in Pakistan in 2003, people started wondering, “Wouldn’t all our problems be solved if they just used Internet Voting?” People all over the world soon started taking a hard look at their voting equipment and procedures, and trying to figure out how to improve them. There is a strong inclination towards moving to Remote Internet Voting – at least among the politicians – in order to enhance voter convenience, increase voter confidence and voter turnout. However, as will be seen later in this independent study, there are serious technological and social aspects that make Remote Internet Voting infeasible in the visible future. Therefore, many technologists have suggested that remote poll-site electronic voting, where the voter can vote at any poll-site (not only his home county poll-site), seems to be the best step forward as it provides better voter convenience, but at the same time, does not compromise security. This report presents a development of the state of the art in Electronic Voting, including the various works done in electronic poll-site voting.

The state of Pakistan emerged from the partition of the Indian sub-continent in 1947. Originally created to meet the demands of Indian Muslims for their own homeland, it split in two in 1971 when the predominantly Bengali speaking eastern part of the country seceded with help from India, to become Bangladesh. With a population of 142 million, it is the seventh largest country in the world. Since independence in 1947, alternating periods of civilian and military rule and human rights abuses have undermined political and economic stability in Pakistan. During the last few decades’ corruption, inefficiency and confrontations between internal institutions have tarnished civilian politics and there has been a series of military coups.

The most recent of these occurred in October 1999 when the elected government of Prime Minister Nawaz Sharif was overthrown in a bloodless coup by General Pervez



Musharraf. In accordance with this judgment, on 14th August 2001, General Musharraf announced a 'roadmap for the restoration of democracy' indicating that elections to the Provincial and National Assemblies as well the Senate would be held in October 2002. Prior to this, on 30th April 2002 President Musharraf held a referendum seeking endorsement for an extension of his rule for a further five years.

## 1.1. ELECTORAL SYSTEM

In early 2002, the military government suggested revised electoral systems to both the Parliament and the Provincial Assemblies. After consultations with civil society organizations, political parties and other stakeholders General Musharraf finally announced the changes to the electoral system on 21 August 2002 as the Legal Framework Order, 2002 was introduced. It was also announced that elections to the National and Provincial Assemblies would take place on 10 October 2002 while elections to the Senate were re-scheduled for 12 November 2002.

The more salient changes were:

- Number of seats in the National Assembly increased; □
- Number of seats in the Senate increased;
- Number of seats in the Provincial Assemblies increased;
- Set aside seats for women were introduced in both houses of the Parliament and the Provincial Assemblies;
- Separate electorate for the minority seats was abolished and a joint electorate
- Voting age was reduced from 21 years of age to 18.

The electoral system of Pakistan reflects the federal system laid out in the 1973 Constitution. As a result, the provinces have a significant impact as how the Senate is composed, which is the norm in most federations.

All the three assemblies have a mix of directly and indirectly elected representatives. A vast majority of the members of the National Assembly and the Provincial Assemblies



are directly elected (79% and 78% respectively) while the opposite relationship can be found in the Senate (92% indirectly elected by the Provincial Assemblies and 8% directly elected by the FATA electorate)

## 1.2. NATIONAL ASSEMBLY

The National Assembly has three categories of seats; (1) general seats, (2) women seats and (3) non-Muslim seats. It should be stressed that only the general seats are directly elected via the First-Past-The-Post system, i.e. by simple majority in single member districts. The women seats, on the other hand, are indirectly elected using a proportional system based on the number of general seats won by the each political party from the Province concerned in the National Assembly. The non-Muslim seats, on the other hand, are indirectly elected via the same proportional electoral system as used for the women seats, except that the entire country constitutes the constituency. Both women and non-Muslims are picked from the respective closed party list. As a result, independent candidates could only run for the general seats and not for any of the reserved seats in the National Assembly.

A further novelty in this year's election is the five per cent threshold introduced to prevent smaller parties access to the reserved seats for women and non-Muslims. If a party doesn't receive at least five per cent of the general seats in the National Assembly, it will not be allotted any of these reserved seats to the National Assembly.

The number of seats in the National Assembly (NA) increased significantly. In 1997 elections, 217 members of the NA were elected (directly and indirectly) while this time around the figure had risen to 342 (see Figure(1)). One of the main reasons for this increase was the introduction of dedicated seats for women (60), but also the general seats increased by sixty-five new seats. Also after the increase in the number of seats in the National Assembly, Punjab will still be the key province since it holds more than fifty per cent of the seats



Category of Seats	1997 Elections	2002 Elections
General Seats	207	272
Women Seats	0	60
Non-Muslims Seats	10	10
Total No. of Seats	217	342

**Figure (1.1) National Assembly Composition: 2002 elections compared to 1997**

### **1.3 THE CHARACTERISTICS OF A GOOD ELECTRONIC VOTING SYSTEM**

The characteristics of a good electronic voting system will, of course, depend on the purpose for which the system will be used. However, there are enough similarities between most polls that it is possible to develop a set of general characteristics that are likely to be desirable in most situations. Indeed if you think about the various elections in which you have participated, you will probably realize that these polls have several common elements.

Prior to an election, organizers determine who is eligible to vote. This may involve a formal registration period or an announcement that anyone who is a member of a certain group as of a certain time may vote. Once the election begins, administrators may validate the credentials of those attempting to vote. This may involve asking voters for identification cards or passwords. Generally, this procedure also involves keeping track of who has already voted so that eligible voters may vote only once. After validating each voter, the administrators collect the voted ballots. Finally, the voted ballots are tallied to determine the election result. Thus, a typical election involves registration, validation, collection, and tallying. Because the registration task often takes place prior to the actual election and may employ techniques currently used for traditional elections, this article will focus on the other three tasks.





To have confidence in the election results, people must believe that the election tasks are performed properly. However, there are numerous opportunities for corruption during the performance of each of these tasks. For example, election authorities may cheat by knowingly allowing ineligible voters to register, allowing registered voters to cast more than one vote, or systematically miscounting or destroying ballots. In addition, ineligible voters may register (often under the name of someone who is deceased) or eligible voters may register under multiple names. Registered voters (eligible and otherwise) may also be impersonated at the polls, and ballot boxes, ballots, and vote counting machines may be compromised.

Traditionally, election fraud has been prevented through the use of physical security measures, audit trails, and observers representing of all parties involved. But the prevention of election fraud is made more difficult by the frequent requirement that votes remain private. Observers may not observe a ballot until after it has been placed in a ballot box, and audit trails must not provide the ability to link a ballot back to the voter who cast it. Even so, these security measures generally work well enough that the possibility of widespread fraud is small and people have confidence that election results are accurate.

When designing an electronic polling system, it is essential to consider ways in which the polling tasks can be performed electronically without sacrificing voter privacy or introducing opportunities for fraud. In order to determine whether a system performs these tasks well, it is useful to develop a set of criteria for evaluating system performance. The following is one set of desirable characteristics for electronic polling systems which incorporates the characteristics of most systems described in the electronic voting literature:

**Accuracy.** A system is accurate if

(1) it is not possible for a vote to be altered



(2) it is not possible for a validated vote to be eliminated from the final tally

(3) it is not possible for an invalid vote to be counted in the final tally.

In the most accurate systems the final vote tally must be perfect, either because no inaccuracies can be introduced or because all inaccuracies introduced can be detected and corrected. Partially accurate systems can detect but not necessarily correct inaccuracies.

**Democracy.** A system is democratic if,

(1) it permits only eligible voters to vote.

(2) it ensures that each eligible voter can vote only once.

**Verifiability.** A system is verifiable if anyone can independently verify that all votes have been counted correctly.

**Convenience.** A system is convenient if it allows voters to cast their votes quickly, in one session, and with minimal equipment or special skills.

**Flexibility.** A system is flexible if it allows a variety of ballot question formats, including open ended questions. Flexibility is important for write-in candidates and some survey questions. Some cryptographic voting protocols are inflexible because they only allow for single-bit (yes/no) votes.

**Mobility.** A system is mobile if there are no restrictions (other than logistical ones) on the location from which a voter can cast a vote.

One of the reasons people are interested in electronic voting systems is that they can be mobile. Voter participation might increase if people could easily cast votes from computers in their homes, offices, schools, and libraries. Of course, for governmental elections it would be essential to retain centralized polling places for people who would not otherwise have access to computers.



The mobility property itself is a major contributor to some of the problems associated with designing a secure and private electronic voting system. By allowing voters to cast their votes from virtually anywhere, we dramatically expand the universe of ineligible people who may attempt to vote. We also limit our abilities to prevent voters from proving how they voted, as there are no longer private voting booths that can prevent vote buyers from observing vote sellers as they cast their votes.

## **1.4 SUMMARY**

This chapter gives a picture of present manual based Election System, clearly defining their drawbacks and the approaches that will cater the ambiguity of the present scenario. These characteristics are the main features of any reliable e-Election System.



## CHAPTER 2

---

### 2. EXISTING PROBLEM OF ELECTION SYSTEM

The present election system is based on manual procedures. The process is quite primitive and has certain flaws in it. These include:

#### 2.1 CORRUPTION

##### a. Duplicate voting

One person is allowed to vote only once. However, it has been happening that one person votes for more than one times, hence helping the candidate win (unfairly).

##### b. Unsatisfactory Counting Procedures

The counting is manual and has been complained a lot of times as being unfair and biased.

#### 2.2 RESOURCES

##### a. Money

Lot of money is wasted on printing and developing of the ballot papers. Furthermore, this is not a one-time expense, since the candidates wont be same every elections.

##### b. Human

A great deal of human resource is required to conduct the elections. This is because several individuals are required to perform each of the tasks (including identification of the voter and allotment of the slips).

##### c. Time



The voting process is very slow. The voter has to go to various desks for different procedures. This makes the process time inefficient. For this reason, all the voters usually don't get the chance to vote in the time period (9:00 am to 5:00 pm).

## **2.3 MAINTENANCE**

### **a. Maintenance of Records**

Maintenance of records is difficult. The voter lists are manual and may contain certain errors. The data contained sometimes is redundant and inconsistent.

### **b. Manual Procedure Prone to Errors**

The manual procedures sometimes result in errors like data inconsistency. Moreover, the manual counting is not much reliable either.

### **c. Slow Results**

Since the results require counting to be done manually, it takes a greater amount of time to get the results. Hence, the results generation process is very slow.

## **2.4 SUMMARY**

The best way to remove the flaws in the present election system is to devise a computerized election system under the framework of Software Engineering. This will improve the system's efficiency and effectiveness both.



### **3 ANALYTICAL DESCRIPTION OF THE SYSTEM**

The proposed e-Election System is corruption free i.e. one person can only vote once during election. In the proposed system the voting process will be very fast and the voters does not have to go to various personals for different procedures. In the computerized Election System a less human resource is required to conduct the election process. The counting process will be fair and unbiased with the help of computers. Hence the proposed system will be cost optimal.

A considerable amount of votes get wasted due to wrong stamping. Our system would provide computerized process so that no vote is wasted. Manual counting is prone to errors and is unsatisfactory. In our automated system, errors like counting mistakes could be eliminated. A lot of resources are needed which include financial resources, human resources and time. Our system would not require large financial resources each time the system is used. The system will only need to be updated.

Maintenance of records is difficult and may result in inconsistencies and redundancy. In computerized database systems, maintenance would be easy since it would only require updating and the database system will take care of inconsistencies or redundancies that may arise. The result generation process is very slow. Our computerized system will generate the result immediately.

#### **3.1 APPROPRIATE MODEL FOR THE ELECTION MANAGEMENT SYSTEM**

Before discussing the various models it is important that a definition is made. The cycle begins with the process of specifying requirements, followed by producing a design, which is then implemented and tested, before being signed off as complete and regarded



as a product. As a product, or through testing, users may find bugs, which are required to be fixed. Software evolves over time and later versions are released which require the phases mentioned above to be re-iterated through.

The requirements specification will describe what the software can and cannot do. It will state the functional and non-functional needs of the software, as specified by the users of the intended software. Along with interface/configuration/software requirements, the specification is used as the input to the design process. With this input a final design will evolve and consist of a system of objects that fulfils the requirements, a description of the public behavior of those objects and the patterns of communication between the objects.

The implementation stage follows the design phase. Evidently if the design is clear then it is relatively easy to implement the solution to meet the requirements. This will involve coding the various components and ensuring that they are able to interact with each other. When testing the finished software it is beneficial if it can be broken into various sub systems, which can be isolated and tested individually. This way error can be traced to their sources easily. With a clear specification as to how the various components interface with each other, it is easy to spot where problems between components exist and their implications. In order to do this, each component's behaviors and actions need to be fully understood.

In order to understand code in the future, clear documentation is required so programmers who look to maintain the software or produce fixes are able to understand where actions occur and how it all fits together. For this reason object-oriented design should be used as this uses encapsulation and information hiding, and communication patterns are rigidly constrained.

After analyzing the project properly, we have come to the conclusion that the model that best suits is “**The Water-fall Model**”. The reasons are as follows:

**Availability of detailed information:** Not just the core requirements, the detailed requirements regarding the Input/Output, processing and storage are known to us.



**Management of technical risks:** From stage One onwards, we can check for technical risks and control them. Hence, the system implementation can be planned to manage the technical skills.

**Iterations not required:** The presence of detailed requirement information allows us to carry out each stage comprehensively, such that nothing is left behind. This reduces the need for any iteration to minimum.

**No time constraint:** We don't need our product to be competing in the market. Rather we are developing the product for an organization, which needs a comprehensive software product to put to work for future elections. Providing prototypes and segments is not appropriate. The project is to be implemented once and for all. We are not time constrained. The requirement is: High quality software.

**Final product must satisfy all requirements:** We require a quality product. The product is completed after passing through various stages of the model, each stage only possible after the completion of the previous stage. Hence, the final product is complete and caters for all the requirements of such a system.

**Conclusion:** The linear sequential model makes our product management easier and can help us better to develop a high quality Election Management System.

## 3.2 FEASIBILITY REPORT

### 3.2.1 Organization without New System

The existing system is paper based with the records of voters and nominees maintained on paper. The voter's identification is done outside the polling booth by the representatives of the election commission. A lot of human resource is required to facilitate the voter i.e. leading the voter to the correct room for voting and providing the voter with the ballot paper etc. The voter stamps on the ballot paper and puts it in the ballot box. Voter is then marked on the thumb so that the voter cannot vote again. After





the polling is over, counting is performed on each polling booth. The results are sent to the central election commission where the results are compiled and later declared. The counting is manual and it takes time for the results to be declared.

### **3.2.2 Business Objectives**

The business objectives of our organization are that the system should be secure, user-friendly, and manageable and time efficient. There would be a proper identification system such that one person can vote only once, making the system secure. The system would provide proper interface, which make it easy for the voters to vote. The computerized database maintained would be making it easy maintain and update the records. The result generation process would be fast since computer will deal with the counting process.

### **3.2.3 Transferring Information To and From Other Processes In The Organization**

The election system will require information of the nominees and the voters. The system can use National Database and Registration Authority (NADRA)'s database. Our system should also be able to transfer information to and from other processes in the organization.

### **3.2.4 Technology Requirement**

We are automating the election system. So we require the computer technologies for the purpose. However, the technologies are currently available in the market, so these could immediately be implemented.

### **3.2.5 Supported and Unsupported Features:**

Our primary features should be maintaining voting lists, candidate lists, maintaining results, voter identification and validation. Our system should be efficient so that more voters can be dealt in a unit time, increasing the voter turnout. For add-on features, we can develop a system which could implement a remote voting system such that a person



can vote for a candidate of his local area from any place. Also we can introduce online voting system using internet and mobiles.

### **3.2.6 Recommendations**

#### **3.2.6.1 Contribution of the New System to the Overall Objectives of the Organization**

The business objectives of our organization are the conduction of secure fair and efficient elections. Our system is providing secure and fair election system, an easy user interface, and a quick voting process and also immediate generation of record. Hence, our system is helping the organization achieve all its organizations business objectives.

#### **3.2.6.2 Interaction of New System to Other Processes**

Since our system can use the database of NADRA, it operates well with the overall system. We can say that it directly interacts and can be integrated with the other systems in organization.

#### **3.2.6.3 Technology Requirements and Constraints**

Our system can be implemented using the resources available (time and cost). However, technological resources are required since we are using a totally new technology. However, in other processes of the organization (NADRA), this technology is being used. So we can easily switch to the new technology.

#### **3.2.6.4 Recommendation**

Taking in view of the above discussion we suggest we continue with the requirement engineering and development process.



### **3.3 SOFTWARE AND HARDWARE REQUIREMENTS**

#### **3.3.1 Hardware Requirements**

- Server - Intel-Pentium 4 machine with 2GHz CPU and 512MB RAM
- Client PC with minimum capability to act as a thin client
- 500 MB of available hard-disk space minimum
- 1024 x 768, 16-bit High Color minimum
- USB Port (for USB peripherals)

#### **3.3.2 Software Requirements**

The main software resources to be utilized for the project include the following.

- SQL Server 2000
- Visual Studio .Net 2003 or 2005
- Windows XP SP2/2000 Professional/2000 Server with SP4
- Crystal Reports v10

### **3.4 SUMMARY**

This chapter deals with the analytical phase of the system, which includes the selection of appropriate model and estimates the feasibility of the overall project. It also specifies the basic hardware and software resources which are required to support the implementation phase.



## **4. SYSTEM LOGIC**

### **4.1 FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS OF ELECTION MANAGEMENT SYSTEM**

#### **4.1.1 Functional Requirements**

- Maintain the category of election (National or Provincial or Union)
- Enter the records of the candidates.
- Enter the records of the voters.
- Maintain the Halqa.
- Interoperate with other organizational systems (NADRA's database).
- Provide voter identification to check if the voter belongs to the specific Halqa.
- Provide voter validation to ensure that the voter has already not voted.
- Provide help facility to the voter.
- Vote casting.
- Selection of election category by voter.
- Provide list of related candidates to the voter.
- Submission of vote by the voter.
- Provide confirmation to the voter.
- Updating voter record (that the voter has caste the vote).
- Update results (increment the count of the candidate to which the user has voted).
- Vote counting.
- Result generation.
- Counting of seats of each party.
- Result display.
- Security.



#### 4.1.2 Non-Functional Requirements

- The category for which the elections are held should only be displayed.
- The voter should only be allowed to vote in his own Halqa.
- The voter list must not have redundant entries .The voters must have NIC as their primary key.
- The candidate list must not have redundant entries. A candidate list must contain only one entry for a particular candidate. However, a candidate's record can be present in more than one Halqa's.
- Voter validation should be performed by checking whether the voter has already voted or not. The voter list record will contain a field (which will be checked if the voter has already voted and unchecked otherwise).
- The system should be simple. Television training will be conducted. Tests will be performed. If the less than 10 out of 100 people whom training is provided, commit errors, than the system will be termed as simple.
- The voter should be allowed to vote for only one candidate in a single category.
- The candidate list viewed by the voter must be self explanatory (providing the candidate sign/symbol).
- The results of the elections should not be displayed until the end.
- The system should perform flawlessly. Testing must be performed. The system must not generate more than 5 errors on average per day.
- The system should be time efficient. Testing should be performed. On average, a voter must take 3 minutes to cast a vote.
- No one should be allowed to access the database of results.



### 4.1.3 Interface Requirements

The noun, interface, is taken to be a discrete and tangible thing that we can map, draw, design, implement, and attach to an existing bundle of functionality. User interface design is the collection of methods and devices that allow interaction between systems and the human beings who use them. They communicate information between the user and the system and vice versa.

Any elements on the screen must be legible and clear to understand – this includes both text and graphics. It is important that numerous users view the design and provide feedback so that it can be improved. The background of the screen should not be busy or patterned as it can cause focus to change from the element in the user's brain. The background should preferably be white, but there is another theory that short wavelength colors such as green should be used because the eye can't observe distracting patterns in backgrounds using these colors.

Color, size and position are factors in drawing attention to elements. Colors that contrast with the background draw the eye's attention. The larger an element is the more users tend to consider it, and therefore more important elements should be larger. Readers read pages from top to bottom and from left to right. Therefore elements in the top left are considered first and so the most important elements should be placed at the top. When images or icons are used they should try not to confuse the user, as they could seem to be a decoration. In order to prevent any confusion they should be accompanied by a caption or label to explain their purpose. Using icons can cause mixed reactions as they can imply a variety of things to different people. In order to counter this text labels should be used. Along with taking these design theories into consideration there are also user interface design principles, which will be abided by. These are:

- The interface should feel familiar to the user and therefore terms and concepts being used should be understood by the users of the system.
- There should be consistency in the way in which operations on the interface are carried out if available on numerous interfaces.
- The user should not be surprised by the actions of the system



- There should be built in mechanisms such that the user will be able to recover from errors.
- There should be context sensitive guidance for users and meaningful feedback from errors should be made available.
- There should be appropriate interaction facilities for different types of users so that the interface is customizable.
- These ideas will be implemented into the final design so that the interface is one which will be easy to use, or easy to understand and without any conflicts.

In this project we will follow these entire interface requirements described above, and will try to design a simple, interactive and user friendly environment of the application. As these are the basic requirements regarding the interface, so they must be put up with the designing phase.

## **4.2 SUMMARY**

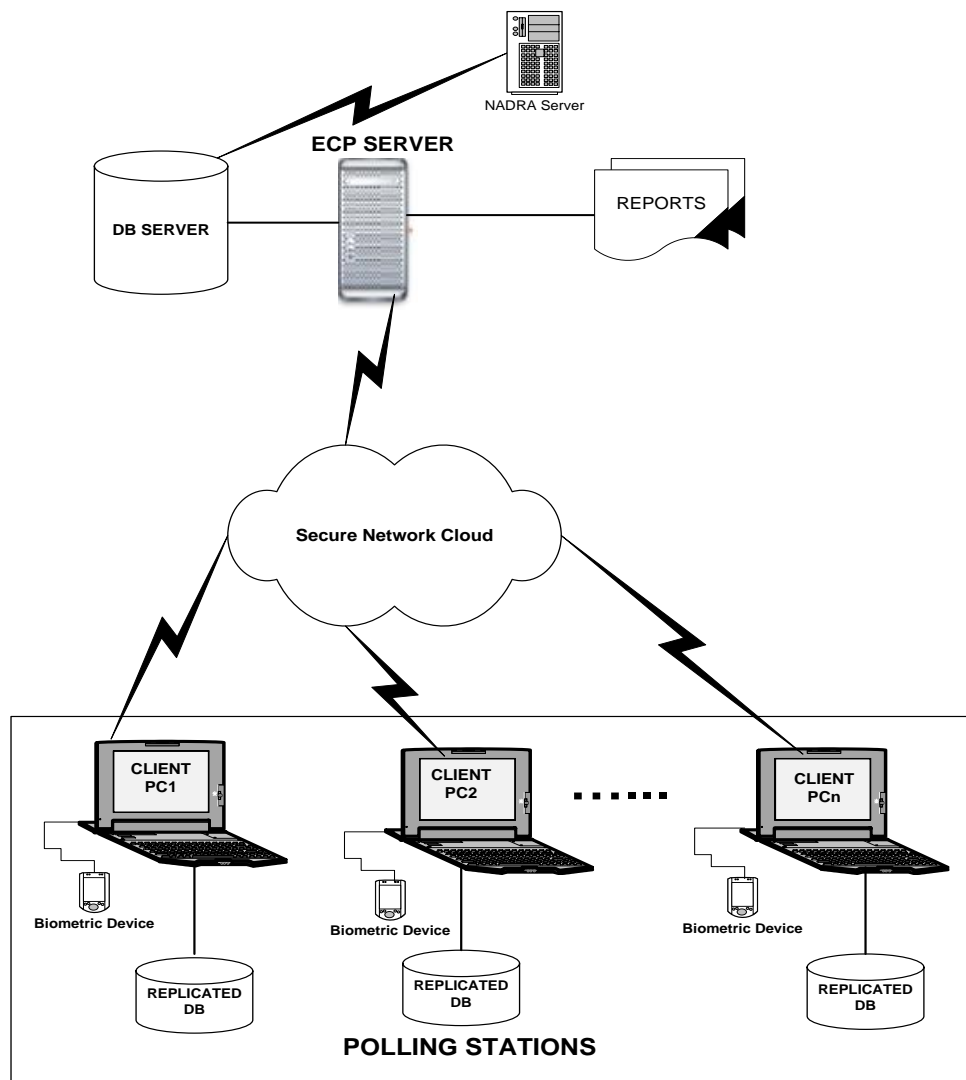
This chapter explained the logic behind the Election System and defined a set of requirements for each component of the system and depicts the clear specifications of functional, non-functional and interface requirements.



## 5. SYSTEM ARCHITECTURE

### 5.1 ARCHITECTURE OF THE SYSTEM

Figure(5.1) shows the overall architecture of the system, and with which components user will interact with the system.



**Figure(5.1) System Architecture**





## 5.2 ADMINISTRATION

The server enrolls the voter in the system; this includes his NIC, bio data and his fingerprint templates. For the security reasons these pre stored templates are used to match with the fingerprint at the time of vote casting.

The server also maintains the information of the Halqa of each city within the country. Information of certain candidates belonging to different parties as electives for the seats of National Assembly and Provincial Assembly are available in the system.

The server also handles the results of the election by generating the reports and displaying it on website.

## 5.3 CLIENT

The client module is deployed at each polling station located within the Halqa. This client module provides the interface to the voter through which he/she can cast vote to their desired party without bearing any type of pressure. The client module is designed to provide a user friendly environment to the voters so that the voters do not find any difficulty in casting their votes.

## 5.4 THE DATA MODEL

The most abstract level of a database design is the data model, the conceptual description of a problem space. Data models are expressed in terms of entities, attributes, domains, and relationships.



### 5.4.1 Entities

An entity is anything about which the system needs to store information. When you begin to design your data model, compiling an initial list of entities isn't difficult. When you (or your clients) talk about the problem space, most of the nouns and verbs used will be candidate entities. "Customers buy products. Employees sell products. Suppliers sell us products." The nouns "Customers," "Products," "Employees," and "Suppliers" are all clearly entities.

The table below shows the description of the entities used in the ER diagram.

ENTITY NAME	Description
Voters	Contains details about the voters.
Voter_Identity	Contains the Thumb Impression for the Voters.
Halqa	Contains Polling Stations within certain areas.
Candidates	Contains details about the candidates for seats of National/Provincial assemblies belonging to particular parties.
Party	Contains list of political parties involved in the Election process.
Vote_Register	Contains the detail about the vote secured by the particular candidate during the polling.
Seat	Contains the details of the seats for particular category i.e. National/Provincial.

**Figure(5.2) Lists of Entities**

### 5.4.2 Attributes

Your system will need to keep track of certain facts about each entity. These facts are referred to as the entity's attributes. Determining the attributes to be included in your model is a semantic process. You must make your decisions based on what the data means and how it will be used.



ENTITY NAME	ATTRIBUTES
Voters	<u>VoterNIC</u> {PK}, <u>HalqaNo</u> {PK}, Name, FatherName, Gender, Address, City, Flag
Voter_Identity	<u>VoterNIC</u> {PK}, <u>HalqaNo</u> {PK}, Template1, Template2, Template3
Halqa	<u>HalqaNo</u> {PK}, Location, City, NIC_REGISTERED
Candidates	<u>CandidateID</u> {PK}, <u>PartyID</u> {FK}, <u>HalqaNo</u> {FK}, CandidateName
Party	<u>PartyID</u> {PK}, PartyName, Intekhaabi_Nishan
Vote_Register	<u>PartyID</u> {PK}, <u>HalqaNo</u> {FK}, <u>SeatID</u> {FK}, VoteCount
Seat	<u>SeatID</u> {PK}, SeatName

**Figure(5.3) Lists of Attributes**

### 5.4.3 Relationships

In addition to the attributes of each entity, a data model must specify the relationships between entities. At the conceptual level, relationships are simply associations between entities. The statement "Parties comprises of Candidates" indicates that a relationship exists between the entities Party and Candidates. The entities involved in a relationship are called its participants. The number of participants is the degree of the relationship.

The relationship between any two entities can be one-to-one, one-to-many, or many-to-many. One-to-one relationships are rare, most often being used between supertype and subtype entities. The relationships between the entities are shown in figure (2) under **section 5.4.4.**

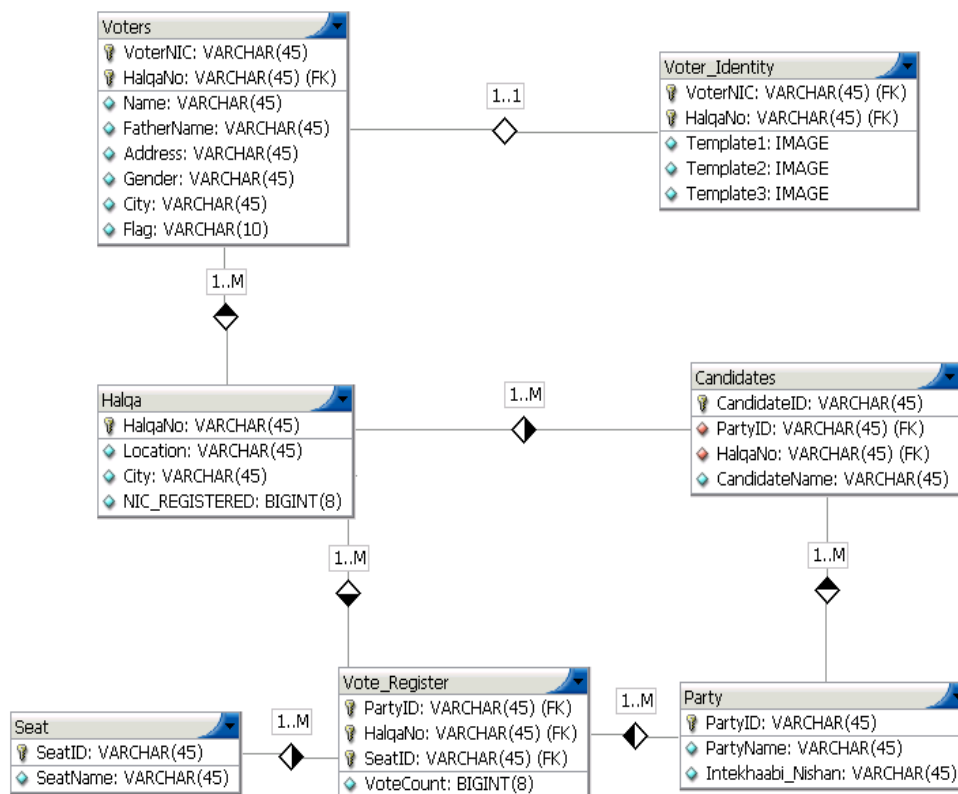


#### 5.4.4 Entity Relationship Diagram of Database

After completing the initial analysis of the system, you'll probably have a set of Entity Relationship (ER) diagrams, a listing of domains used in the system, and some notes regarding data constraints. It's simple to get these organized into a presentable format. The entity analysis, perhaps illustrated by an appropriate ER diagram for complex entities, is adequate documentation of the model.

The Entity Relationship model, which describes data in terms of entities, attributes, and relations, was introduced by Peter Pin Shan Chen in 1976. At the same time, he proposed a method of diagramming called Entity Relationship (ER) diagrams, which has become widely accepted. ER diagrams use rectangles to describe entities, attributes, and diamonds to represent relationships, as shown in Figure (5.4).

The nature of the relationship between entities (one-to-one, one-to-many, or many-to-many) is represented in various ways. A number of people use 1 and M or 1 and  $\infty$  (representing infinity) to represent one and many.



Figure(5.4) Entity Relationship Diagram



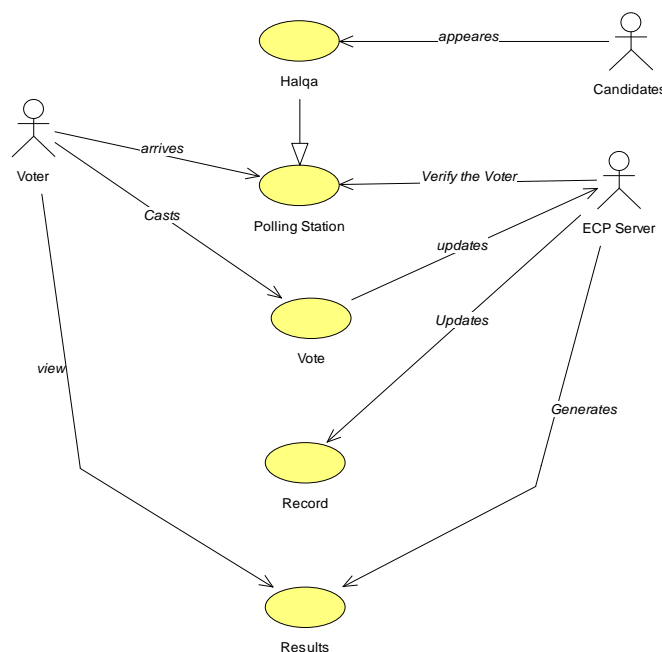
## 5.5 UNIFIED MODELING LANGUAGE

There are a wide variety of different design techniques available today. Due to software components being based in object-orientation, it makes sense to use an object-oriented approach to design as opposed to another method. Unified Modeling Language (UML) is used for visualizing complex object-oriented software and is the perfect choice for modeling this system.

There could be various different solutions to the requirements but this design document will only highlight one possible solution.

### 5.5.1 USE CASE DIAGRAM

Use cases are a good way of getting an overall picture of what is planned to happen in a new system (Schaum's Outlines, UML Bennett, Skelton and Lunn 2001). Use cases display graphically what the various uses of the system are and what other systems or users are involved in the process. From the use cases it is easy to see what high level functionality the component will possess and what roles the various systems surrounding it will perform.

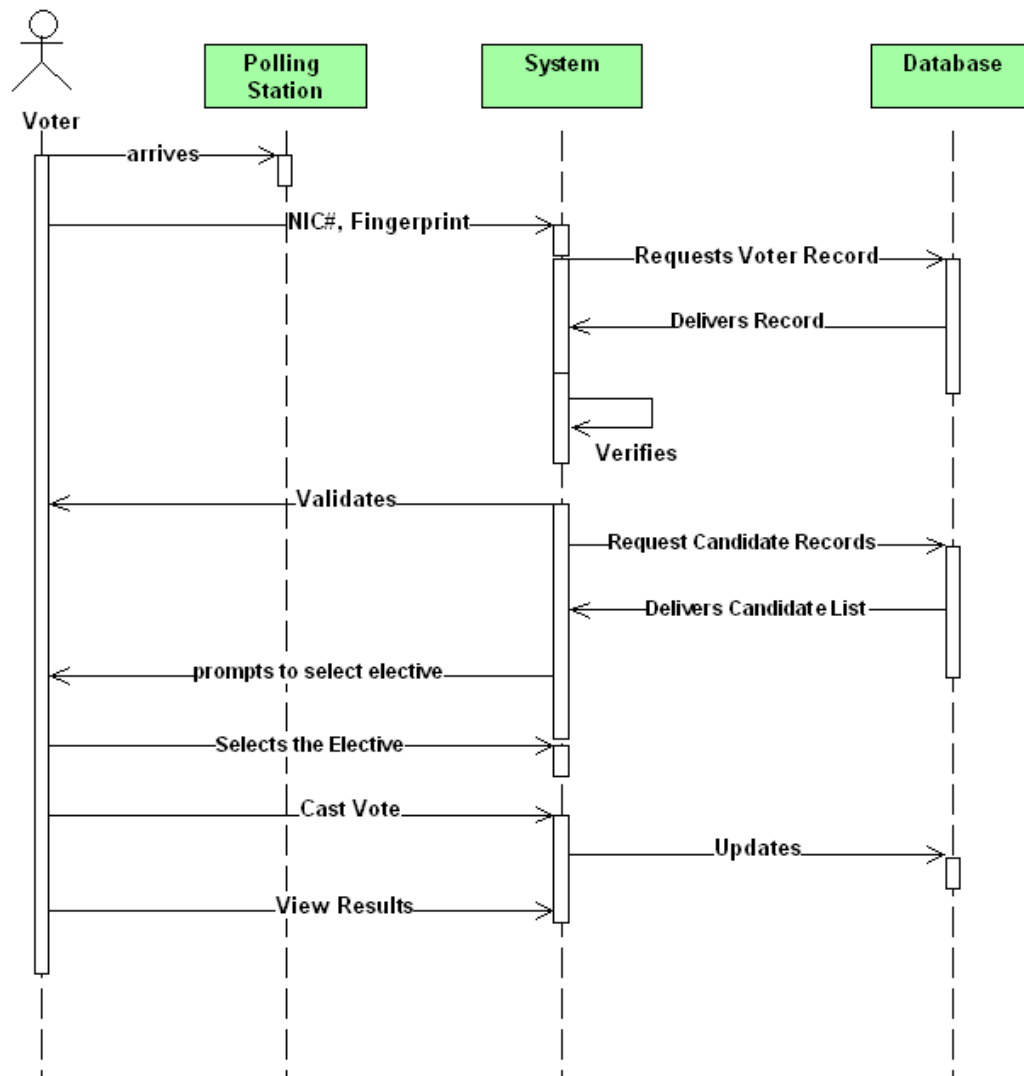


**Figure (5.5) Use Case Diagram**



### 5.5.2 SEQUENCE DIAGRAM

A sequence diagram shows how the various methods and constructors work within a system as an operation is being performed. The operations chosen for sequence diagrams are based on the use cases.



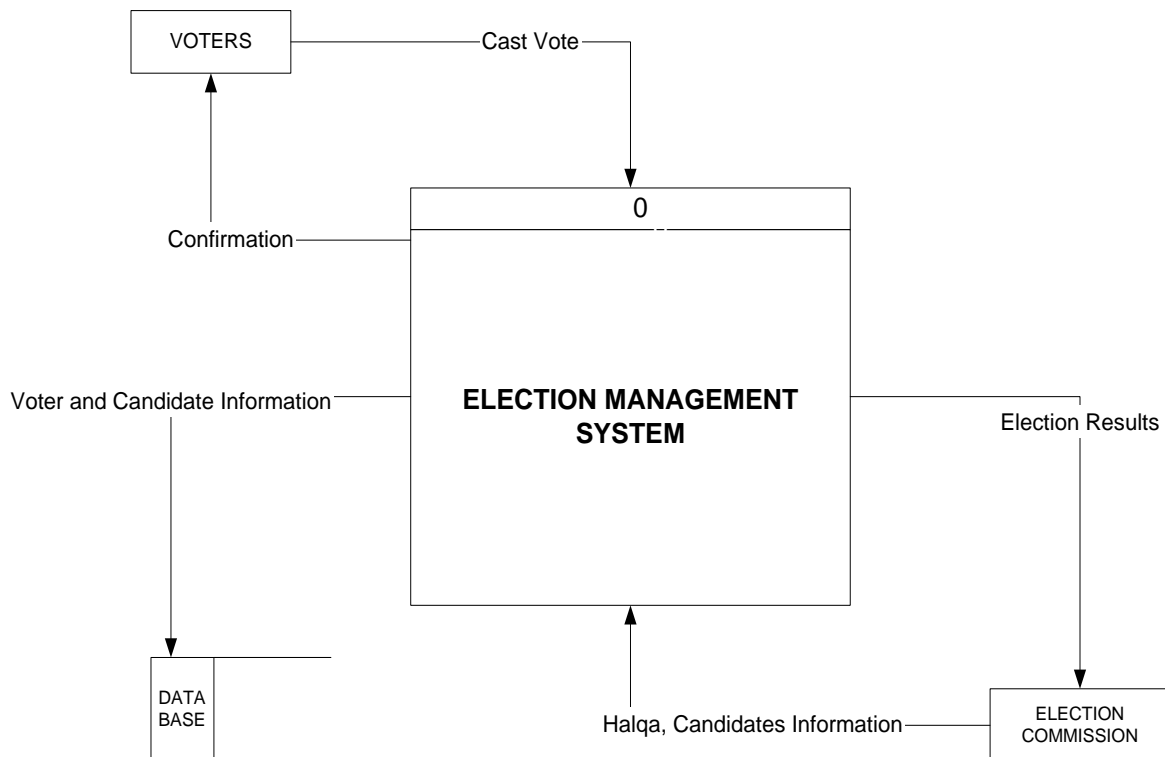
Figure(5.6) Sequence Diagram



## 5.6 DATA FLOW DIAGRAM OF THE SYSTEM

The purpose of a data flow diagram is to show how the data within a system or subsystem will be obtained, transferred or manipulated. These diagrams are able to help distinguish the functions that are required and their complexity and have been generated by analyzing the requirements and observing the users in the tasks that will be implemented. In the diagrams there are a variety of shapes used to demonstrate differing inputs, processes and outputs. In the following diagrams data sources and syncs are represented by parallel lines, roles by rectangles, processes by rectangles with curved vertices, decisions by circles with question above it and data flows are represented by text above an arrow between two shapes. These standards have been enforced into all the diagrams and provide a level of consistency.

### 5.6.1 Context Diagram



**Figure (5.7) Context Diagram**



### 5.6.2 Level-0 Diagram

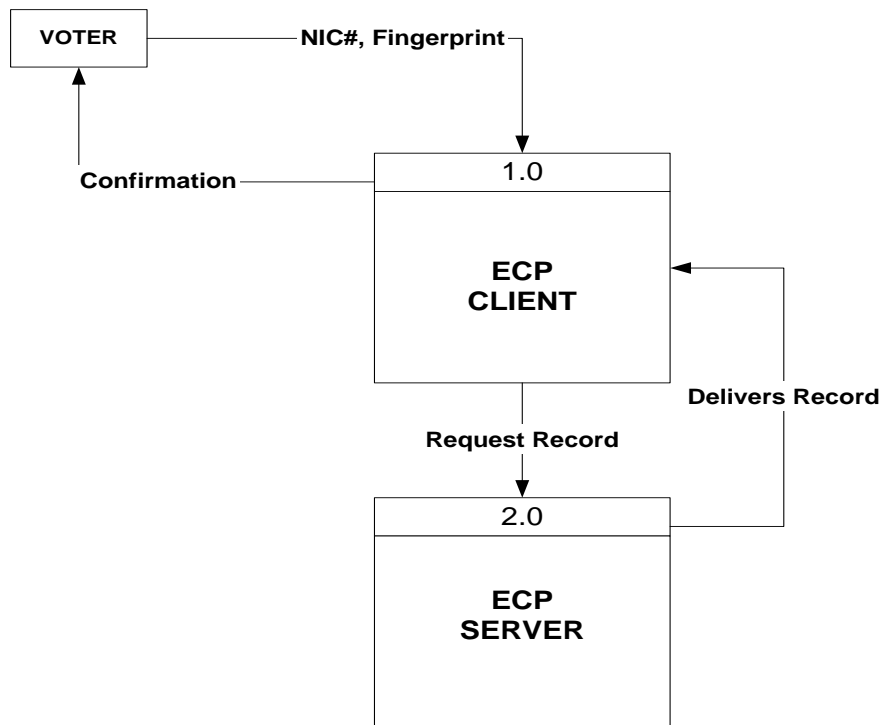


Figure (5.8): Level-0 Diagram

### 5.6.3 Level-1 Diagram

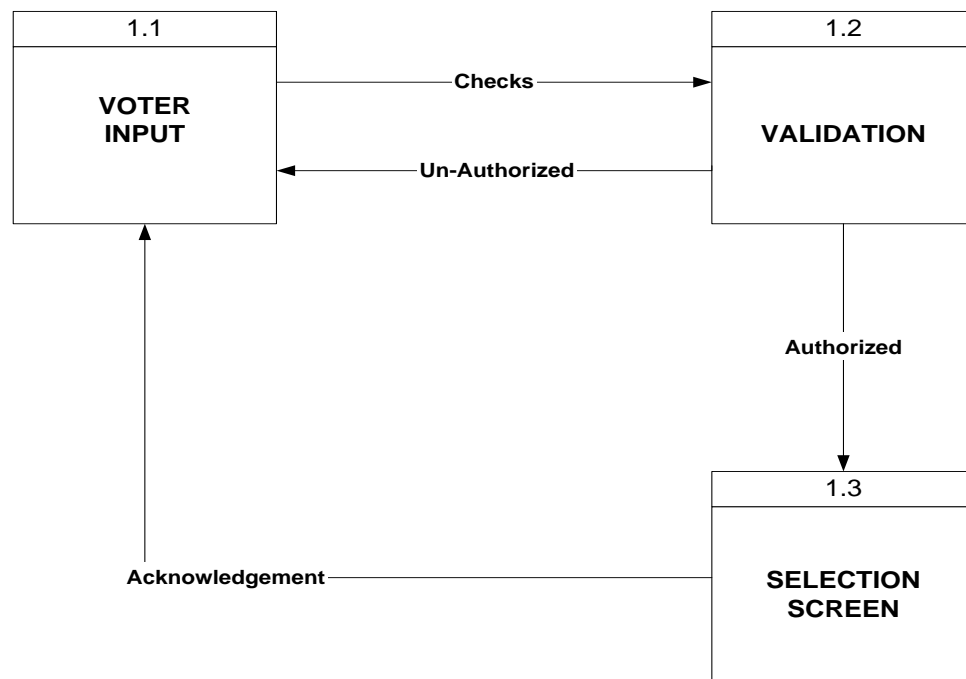
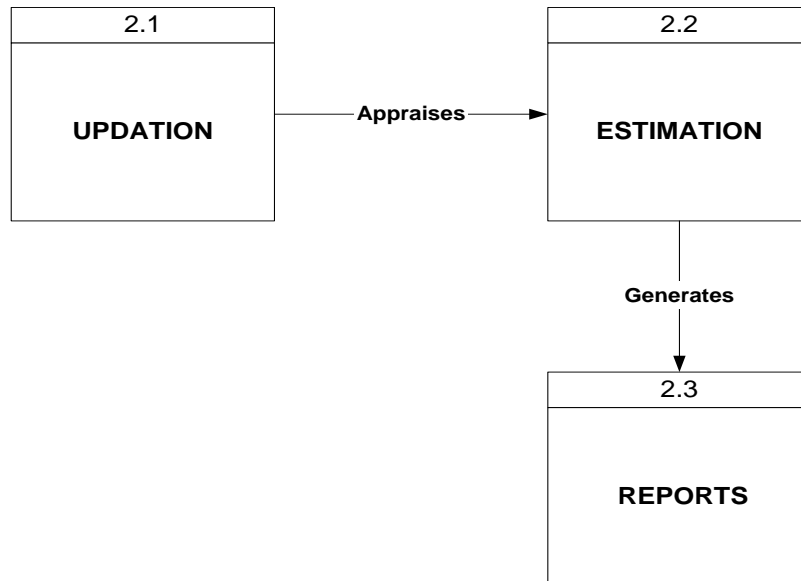


Figure (5.9) Level-1 Diagram





### 5.6.4 Level-2 Diagram



**Figure (5.10) Level-2 Diagram**

## 5.7 SUMMARY

This chapter shows how the components work together in a system architecture diagram. It also describes the system architecture and the data models comprises of ER diagram and data flow diagram in addition to that, it also contains UML diagrams such as use case and sequence diagram which clearly illustrates the overall system.



## 6. ADMINISTRATION IMPLEMENTATION

### 6.1 FUNCTIONAL UNITS

<b>Function name:</b>	Maintenance of voter record.
<b>Description:</b>	We need a database with complete record of voter. It is required for voter identification and validation. Also, we need it for insertion, updating and deletion of voter records.
<b>Input:</b>	NIC#, Name, Address, halqa,
<b>Source:</b>	Database and Election Commission (halqa) .
<b>Output:</b>	Voter list.
<b>Destination:</b>	Voter identification, validation, vote casting
<b>Pre-conditions:</b>	We require a database design containing voter data. It must be in database.
<b>Post-conditions:</b>	The database must remain consistent with the NADRA database. So, if any insertion, updating or deletion is done in the NADRA database, it must show-up also in our database. It should also check for any errors or redundancy.
<b>Function name:</b>	Maintenance of the candidate record.
<b>Description:</b>	We require a database with a complete record of the candidates taking part in the elections. It will maintain the candidate record produced by the Election commission. It will be used for candidate identification.
<b>Input:</b>	Name, Halqa No, Party Name.
<b>Source:</b>	Election Commission.
<b>Output:</b>	Candidate list.
<b>Destination:</b>	Voter screen (providing options to voter to cast vote)



<b>Pre-conditions:</b>	Database design is required to hold candidate data.
<b>Post-conditions:</b>	The database must remain consistent with the NADRA database. So, if any insertion, updating or deletion is done in the NADRA database, it must show-up also in our database. It should also check for any errors or redundancy.
<b>Function name:</b>	Maintaining result
<b>Description:</b>	This will update the result as the voter cast his vote. This will also update party and candidate result. Each counter will be incremented.
<b>Input:</b>	Vote, voter record, candidate record.
<b>Source:</b>	Voter record and candidate record, vote.
<b>Output:</b>	Candidate result and party result.
<b>Destination:</b>	Display election statistics and result calculation and generation.
<b>Pre-conditions:</b>	Vote must be cast.
<b>Post-conditions:</b>	Result should be updated. Candidate record (number of votes get) should be updated. Party record must also be updated.
<b>Function name:</b>	Vote counting
<b>Description:</b>	Initially the voter-counter for each candidate must be zero. Each time a vote is cast, the vote-counter of the corresponding candidate must be updated.
<b>Input:</b>	Votes cast by the voters.
<b>Source:</b>	Database to count the votes.
<b>Output:</b>	The number of votes each candidate gets. This will help determine which candidate wins.
<b>Destination:</b>	Displaying the candidate result and the party result.
<b>Pre-conditions:</b>	Vote should be cast.
<b>Post-conditions:</b>	Result of counting should be properly stored. Its better that the results are updated each time vote is cast.



## 6.2 CLASS DIAGRAM



Figure(6.1) Class Diagram for Administration Module




### 6.2.1 Administration Module



**Figure (6.2) Main Form for Admin Module**



**Voters**

  
**ELECTION SYSTEM OF PAKISTAN**  
 SECURE VOTING

INSERT, UPDATE, SHOW ALL | CHECK, GET, REMOVE

Voter NIC: 42101 - 0000043 - 1  
 Name: Kamran  
 FName: Haque  
 Address: Gulshan  
 Gender: ☒ Male ☐ Female  
 City: Karachi  
 Halqa Number: NA-1


VoterNIC	Name	FatherName	Address	Gender	City	HalqaNo
42101-00000	Ali	Akram	M-78 Classic	Male	Karachi	NA-1
42101-00000	Ali	Akram	M-78 Classic	Male	Karachi	PS-1
42101-00000	Amir	Ali	A-342 Uzma	Male	Karachi	NA-2
42101-00000	Amir	Ali	A-342 Uzma	Male	Karachi	PS-1
42101-00000	Shagufta	Saleem	E-23 Ghani C	Female	Karachi	NA-3
42101-00000	Shagufta	Saleem	E-23 Ghani C	Female	Karachi	PS-2

Copyright (C). 2006, All right Reserved

**Figure (6.3) Data Entry Form for Voters**



**Halqa**

  
**ELECTION SYSTEM OF PAKISTAN**  
 SECURE VOTING

INSERT, UPDATE, SHOW ALL | CHECK, GET, REMOVE

Halqa No:

Location:

City:

Registered NIC:

	HalqaNo	Location	City	Registered_NIC
▶	NA-12	Qasimabad	Hyderabad	1500
	NA-14	Chandni Chowk	Hyderabad	1894
	NA-1	G.Jauhar	Karachi	2000
	NA-11	Latifabad	Hyderabad	2000
	NA-5	Nazimabad	Karachi	2000
	NA-9	Steel Town	Karachi	2000
	NA-10	Safura Goth	Karachi	2345
	NA-16	Gharib Abad	Sukkur	2364
	NA-7	Gulberg	Karachi	2398
	NA-19	Queens Road	Sukkur	2432

Copyright (C). 2006. All right Reserved

**Figure (6.4) Data Entry Form for Halqa**



**Party**



**INSERT, UPDATE, SHOW ALL** | **CHECK, GET, REMOVE**

Party ID:

Party Name:

Intekhaabi Nishaan:

PartyID	PartyName	Intekhaabi_Nishan
NAA	National Alliance	Tractor
PPP(P)	Pakistan Peoples Party(Parliamentarians)	Teer
PML(N)	Pakistan Muslim League(Nawaz Sharif)	Sher
MQM	Muttahida Qaumi Movement	Patang
IND-1	Independent	Pana
MMA	Muttahida Majlis-e-Amal	Kitaab
IND-2	Independent	Hammar
PML(Q)	Pakistan Muslim League(Quaid-e-Azam)	Cycle
PML(J)	Pakistan Muslim League(Junejo)	Bus
PTI	Pakistan Tehreek-e-Insaf	Bat


Copyright (C). 2006. All right Reserved

**Figure (6.5) Data Entry Form for Party**





**Vote Registered**



**ELECTION SYSTEM OF PAKISTAN**  
SECURE VOTING

INSERT, SHOW ALL

Halqa Number: NA-1

Party ID: IND-1

Seat ID: MNA

INSERT SHOW ALL


	HalqaNo	PartyID	SeatID	VoteCount
▶	NA-1	MQM	MNA	1000
	NA-1	PML(Q)	MNA	340
	PS-1	PTI	MPA	188
	NA-1	PTI	MNA	26
	NA-1	IND-1	MNA	0
	NA-1	IND-2	MNA	0
	NA-1	MMA	MNA	0
	NA-1	NAA	MNA	0
	NA-1	PML(J)	MNA	0
	NA-1	PPP(P)	MNA	0

Copyright (C). 2006, All right Reserved

**Figure (6.6) Data Entry Form for Vote Registered**



**Candidates**



**ELECTION SYSTEM OF PAKISTAN**  
**SECURE VOTING**

INSERT, UPDATE, SHOW ALL | CHECK, GET, REMOVE

Candidate ID:

Candidate Name:

Halqa Number:

Party ID:

INSERT UPDATE SHOW ALL

	CandidateID	CandidateName	HalqaNo	PartyID
▶	IND-1	Sheikh Rasheed	NA-1	IND-1
	IND-2	Muhammad Khursheed Khan	NA-1	IND-2
	MMA-1	Qazi Hussain Ahmed	NA-1	MMA
	MMA-2	Maulana Fazal-ur-Rehman	NA-2	MMA
	MMA-3	Hafiz Hussain Ahmed	NA-3	MMA
	MMA-4	Aslam Afghani	NA-4	MMA
	P-MMA-1	Muhabbat Khan	PS-1	MMA
	MQM-1	Altaf Hussain	NA-1	MQM
	MQM-2	Ishrat ul Ibad	NA-2	MQM

Copyright (C). 2006. All right Reserved

**Figure (6.7) Data Entry Form for Candidates**



### 6.2.2 Constraints

- Enrollment
- Availability of voter's record.
- Accessibility of Halqa, Candidates, Seats and Party record's.
- Secure administration.
- Error free
- Corruption free
- User friendly interface
- Modular system
- Secure network implementation.

## 6.3 SUMMARY

This chapter has described the complete class hierarchy of admin application and accounts all of the functional units required to carry out the task and the list of constraint that will be mandatory.



## 7. CLIENT IMPLEMENTATION

### 7.1 FUNCTIONAL UNITS

<b>Function name:</b>	Voter identification.
<b>Description:</b>	We need to identify the voter when he/she comes to vote. We need to check if he has a valid NIC# and that the NIC really belongs to him/her.
<b>Input:</b>	NIC#, Fingerprint impression.
<b>Source:</b>	Voter record, Voter NIC.
<b>Output:</b>	Confirmation of voter being eligible to vote using the NIC
<b>Destination:</b>	Vote casting.
<b>Pre-conditions:</b>	Voter record (list) should be maintained.
<b>Post-conditions:</b>	Voter must be allowed to cast vote in case of correct identification otherwise, he must not be allowed to cast vote.

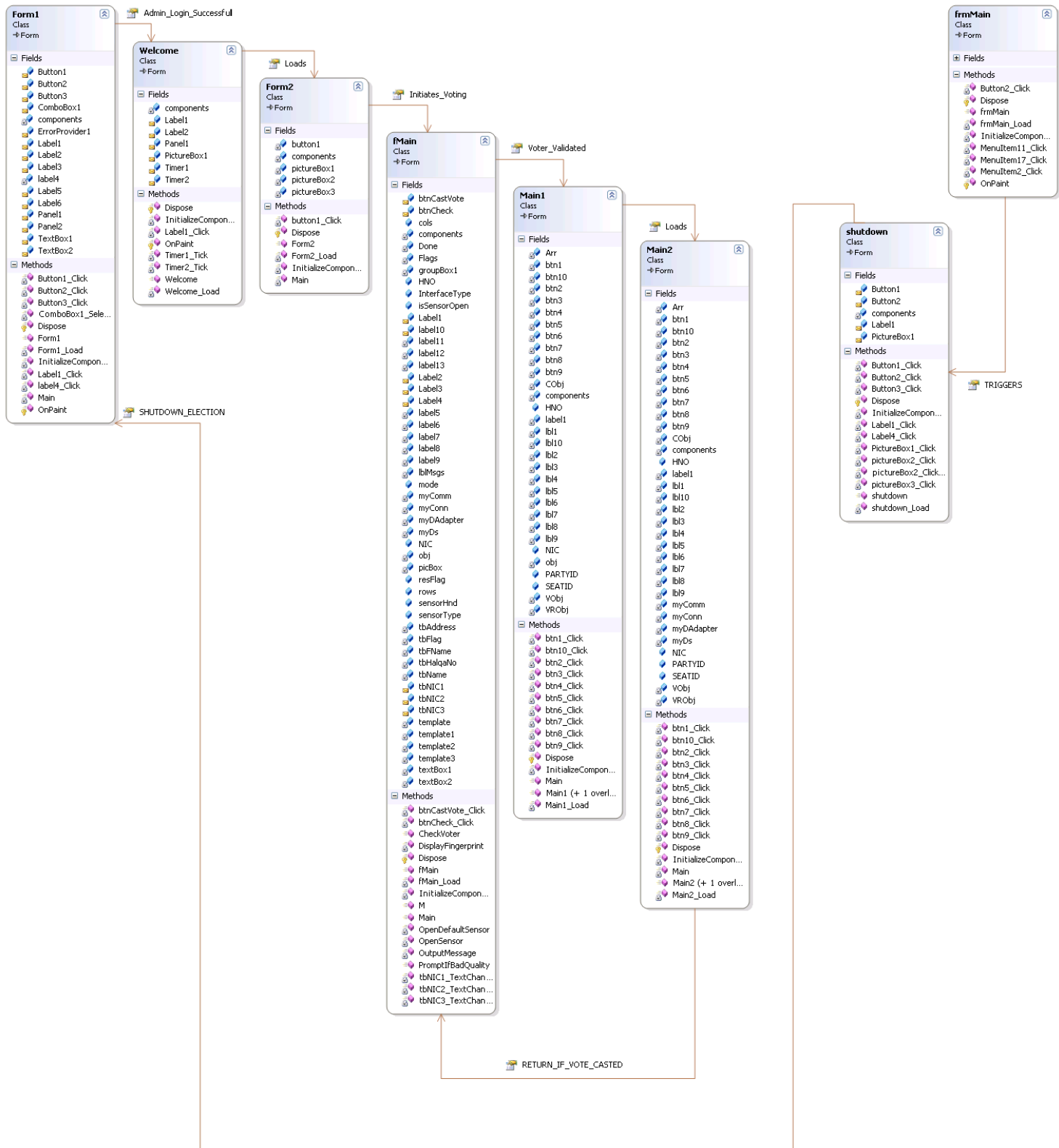
<b>Function name:</b>	Voter validation
<b>Description:</b>	We need to check if a particular voter has already voted or not. Only if he hasn't voted, he should be allowed to vote. So we must maintain a separate attribute which should be checked (true) if the voter has already voted and unchecked (false) otherwise. Using this attribute we can avoid duplicate voting.
<b>Input:</b>	Voter NIC#, Fingerprint impression.
<b>Source:</b>	Voter lists, database.



<b>Output:</b>	On the validation desk, an error message must be placed telling that the voter is not allowed to vote if he has already voted. Otherwise, a confirmation message must be provided.
<b>Destination:</b>	Vote casting
<b>Pre-conditions:</b>	Voter list must be available.
<b>Post-conditions:</b>	Based on the validation result, the voter must be allowed or disallowed to cast vote.
<b>Function name:</b>	Update voter record (vote has been casted or not)
<b>Description:</b>	When the voter comes to cast vote his/her status attribute must be unchecked (false) if he/she hasn't voted already. After casting the vote his/her status should be changed to checked (true) indicating that he/has cast the vote.
<b>Input:</b>	NIC#, vote.
<b>Source:</b>	Voter's record.
<b>Output:</b>	Voter record is updated and candidate counter (votes) is updated.
<b>Destination:</b>	Voter validation.
<b>Pre-conditions:</b>	Initially voter status should be unchecked (false) if he hasn't already voted.
<b>Post-conditions:</b>	Voter status should be changed to check (true) indicating that voter has caste the vote.



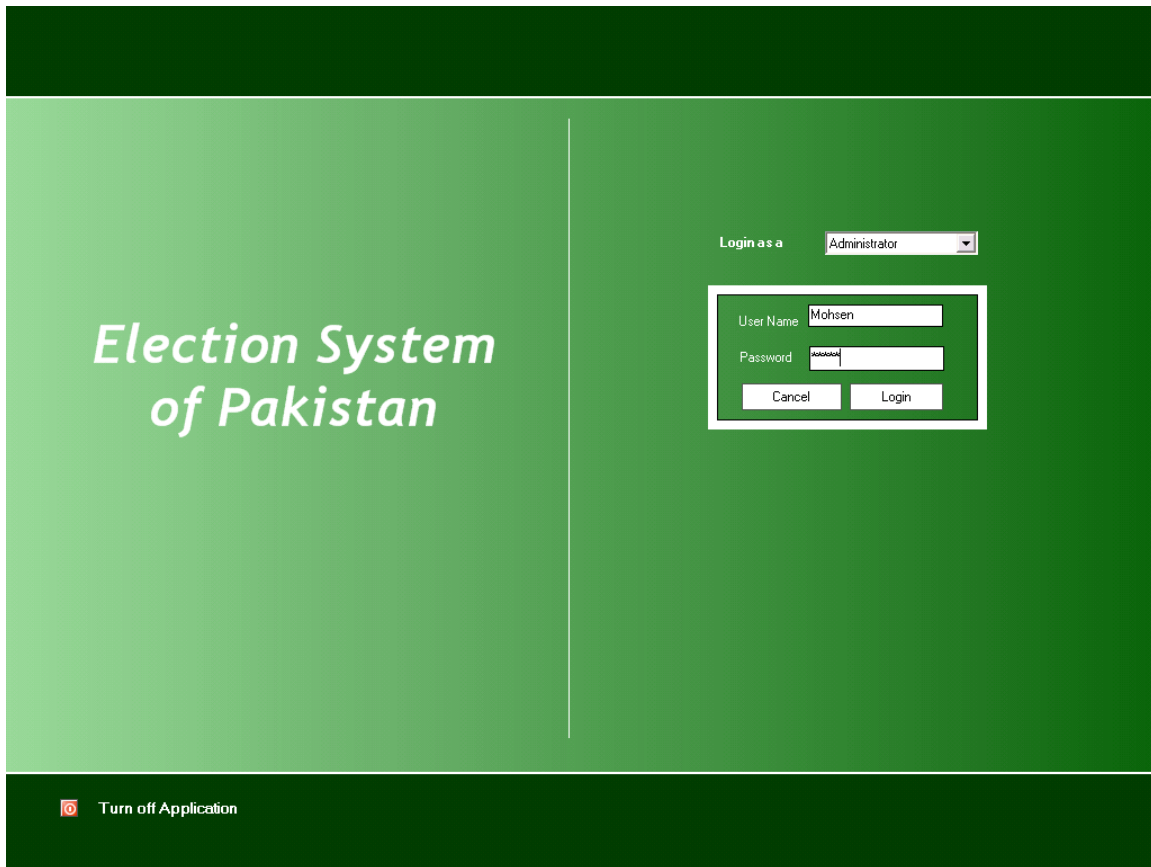
## 7.2 CLASS DIAGRAMS



Figure(7.1) Class Diagram for Client Module



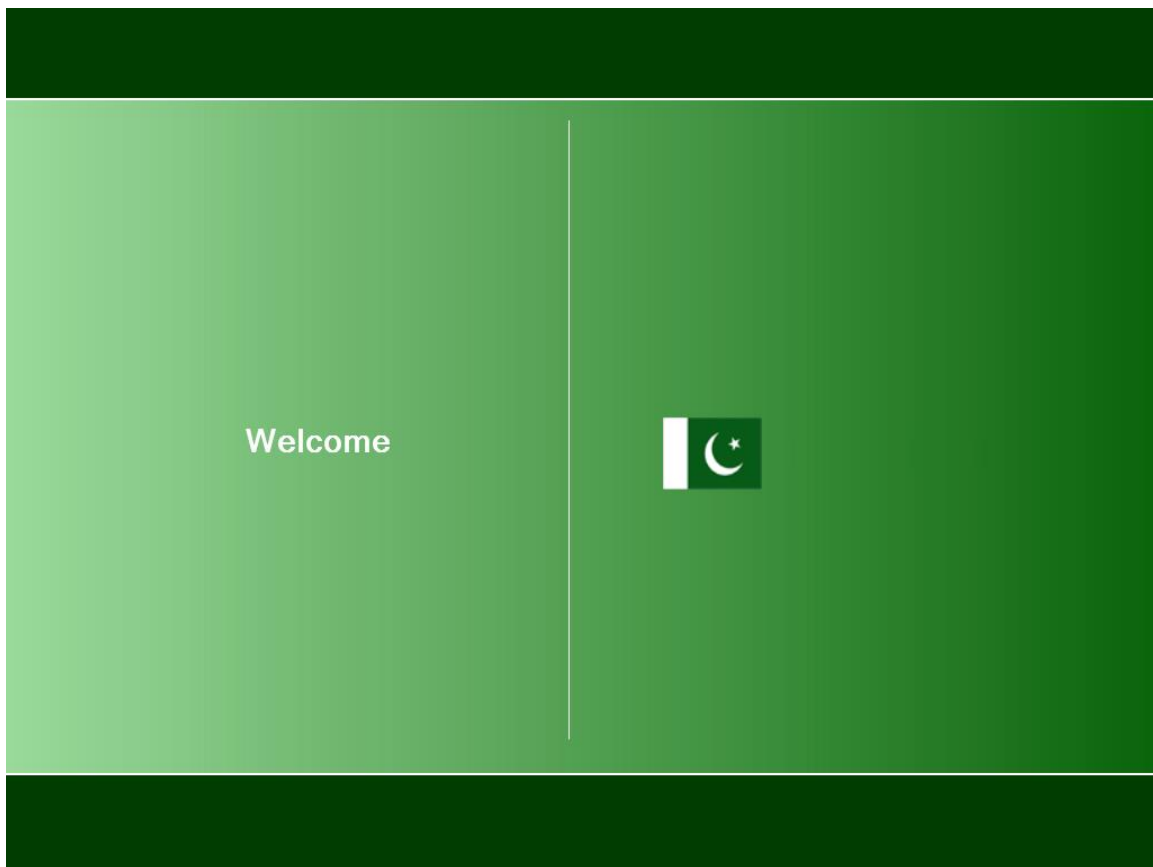
### 7.2.1 Application Forms



The screenshot displays the login interface for the 'Election System of Pakistan'. The background is a dark green gradient. On the left, the title 'Election System of Pakistan' is written in a large, white, italicized serif font. On the right, there is a login form with a white border. At the top of the form is a 'Login as a' label followed by a dropdown menu showing 'Administrator'. Below this are two input fields: 'User Name' containing 'Mohsen' and 'Password' containing 'admin@neda'. At the bottom of the form are two buttons: 'Cancel' and 'Login'. At the very bottom of the screen, there is a dark green bar with a red power icon and the text 'Turn off Application'.

**Figure (7.2) Administrator Login Form**






**Figure (7.3) Welcome Form**






**WELCOME TO ELECTION SYSTEM OF PAKISTAN**

  
**الیکشن کمیشن پاکستان**

Enter Your NIC Number : 42101 - 1111111 - 1 اپنا شناختی کارڈ نمبر درج کیجیے



Voter Detail

Name

Father Name

Address

Halqa No

Copyright (C). 2006, All right Reserved

 Start

**Figure (7.4) Vote Casting Validation Form**





**Figure (7.5) Vote Casting Form for MNA**

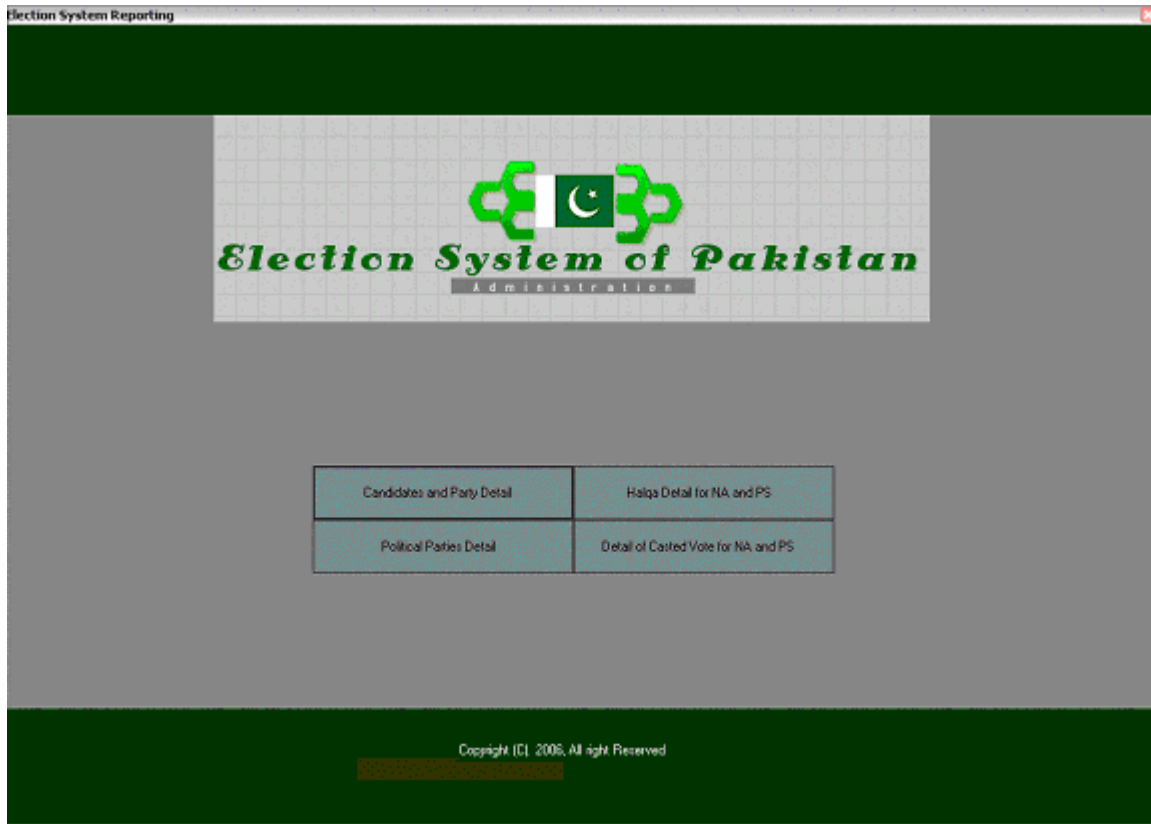




**Figure (7.6) Vote Casting Form for MPA**

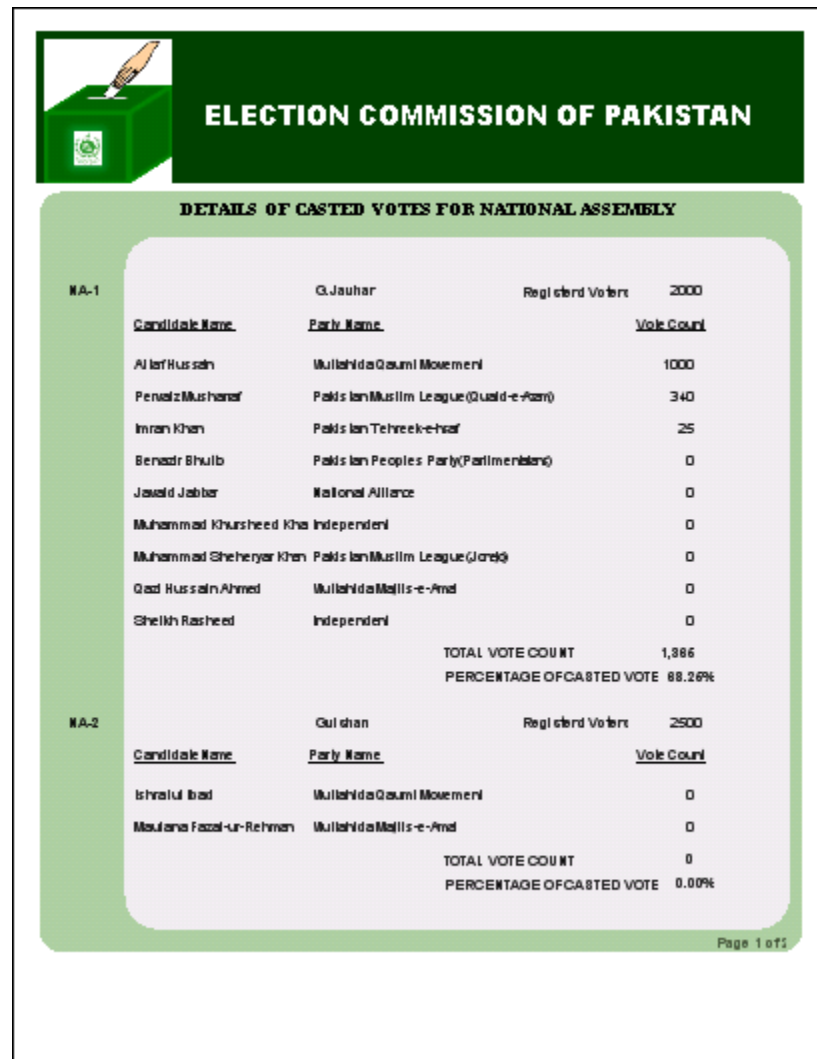


### 7.2.2 Reports



**Figure (7.7) General Report Form**





**Figure (7.8) Election Result Report**

### 7.2.2 List of Constraints

- Security
- Reliability
- Reusability
- Time efficient
- Error free
- Corruption free
- User friendly interface
- Modular system
- Implementation with existing technologies
- Less resource consumption
- Data consistency
- Restrict repeated vote casting
- Restrict a voter to cast within his constituency
- Restrict a voter to cast more than one vote for each category
- Validation of voter through identity card number and scanning of thumbprint

## 7.3 SUMMARY

This chapter has described the complete hierarchy of client application and accounts all of the functions required to accomplish the task and the list of constraint that will be forced.



## **8. TESTING AND EVALUATION**

This chapter describes and discusses both the real-world and black box testing carried out on the Election System, and gives an evaluation of the functionality and user interface of the system.

### **8.1 REAL WORLD TESTING**

This section describes and shows results of the real-world testing procedure carried out on the system.

#### **8.1.1 Procedure**

To test the system in a real-world environment, I went into a practical session consisting of forty five final year Computer Science students. The front end of the system was running on a projected screen at the front of the room.

The session began with a brief presentation of the system, introducing the participants to the Election system to give an insight into the context the system would be used under in a real-world situation.

All participants were asked to complete feedback form. This form includes questionnaires regarding the overall system behavior as they experience.

The questions included in the feedback form are as follows:

- 1) Does the interface seem to look user friendly?
- 2) How easy to use did you find the system to be?
- 3) Does the system provide complete solution to the Election Process?
- 4) Does the response of the system is satisfactory?
- 5) Is there a requirement of the proposed system to be implemented?
- 6) Overall how will you rate the system?(0→5)

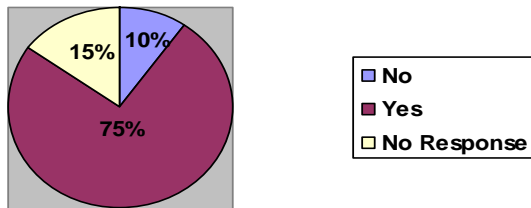


### 8.1.2 Feedback Results

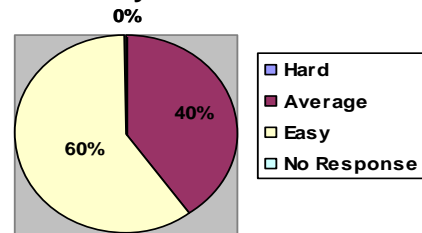
Of the forty five students in the evaluation period, all students have completed the form.

77% (35 students) of the participants were male, 23% (10 students) were female.

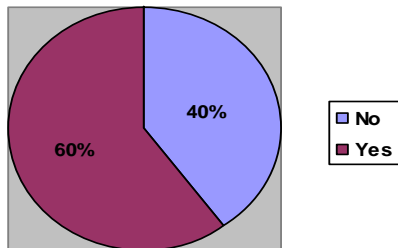
**Does the interface seems to look user friendly?**



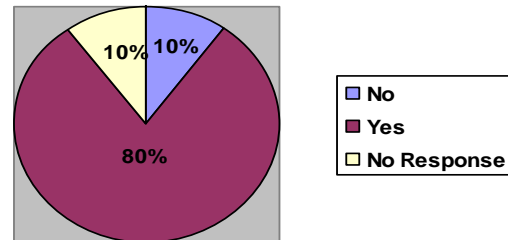
**How easy to use did you find the system to be?**



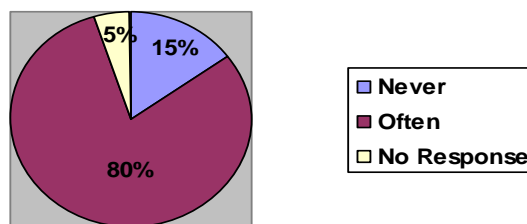
**Does the system provide complete solution to the Election Process?**



**Does the response of the system is Satisfactory?**



**Is there is a requirement of the proposed system to be implemented?**



**Figure(8.1) Evaluation Results**





## 8.2 BLACK BOX TESTING

This phase of testing involves examining each component of the Election system as a black box system. The following sections list the tests carried out, and the expected and actual behavior of the system. Tests were carried out in the network environment having Pentium machines with Microsoft Development Tools.



### 8.2.1 Client Application

TEST	EXPECTED BEHAVIOR	PRECONDITION	ACTUAL BEHAVIOR	RESULT
1). Launch Client Application.	Application starts with Welcome screen	None	As Expected	Pass
2). Enter any invalid NIC#.	Log Error: "You Must Enter Valid NIC #"	1 Pass	As Expected	Pass
3). Leaving Input Blank press Enter.	Log Error: "Enter the Data First"	1 Pass	As Expected	Pass
4). Enter valid NIC#.	Log Prompt: "Place Thumb on the Sensor"	1 Pass	As Expected	Pass
5). Place non aligned Thumb Impression.	Log Prompt: "Place Thumb with proper alignment"	1 Pass	As Expected	Pass
6). Place with proper alignment of Thumb Impression.	Welcome Msg: "You are validated and can cast vote"	1,4, 6 Pass	As Expected	Pass
7). Try to vote two or more candidates.	Log Error: "Invalid Operation"	6 Pass	As Expected	Pass
8). Cast vote to single candidate.	Confirmation Msg: "Your vote has been casted"	6 Pass	As Expected	Pass
9). Terminating the client process.	Client Application terminates and returned to main screen	8 Pass	As Expected	Pass

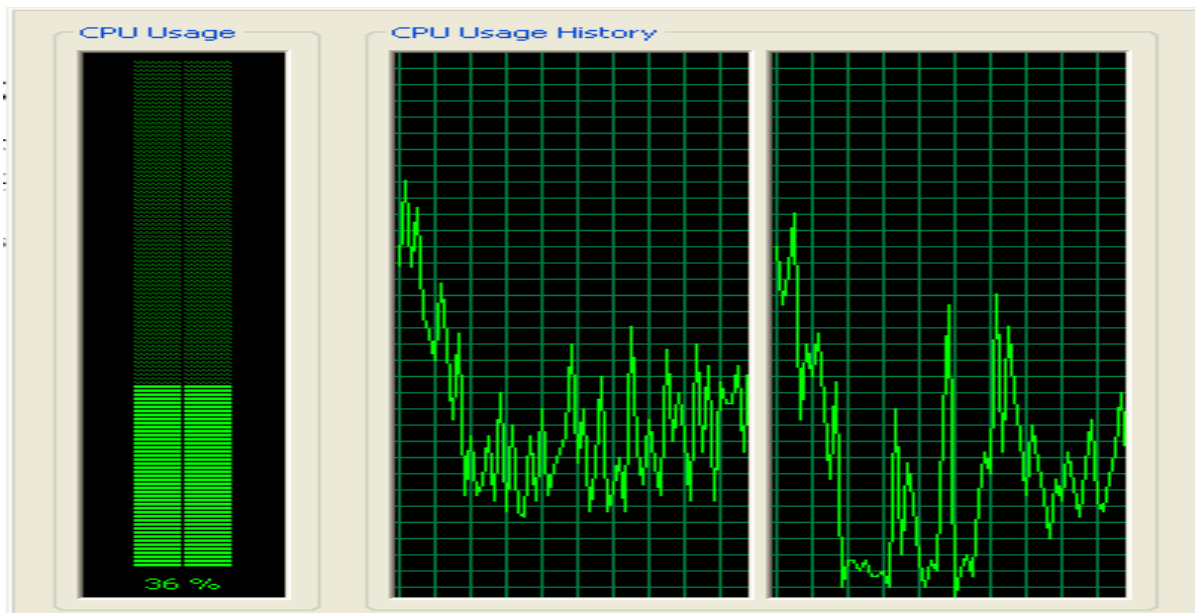
**Figure(8.2) Table showing behavior Testing results**



All test cases for the client application passed without error. Tests 7 and 8 show that the client application correctly prevents multiple voting, even when the voter tries to vote, the system will restrict him from doing this operation.

### 8.3 SYSTEM PERFORMANCE

While executing the front end of the Election System, it has been noted that the performance of the front end is less than ideal. The CPU utilization graphs below show the load on Windows XP based system Figure (8).



**Figure(8.3) Showing the Performance of the System while running Application.**

Figure 8 shows that on the execution of the system, the system utilizes 36% of the CPU, on the Windows XP based system. This is likely to be because of the higher specification system.



## 8.4 SUMMARY

This chapter has described the real-world testing and black box testing carried out on the Election system, evaluated the system against heuristics and design principals, as these were suggested and recommended by IEEE Standard for Software Testing.



## **9. CONCLUSION**

This chapter discusses whether this project has met its objectives and aims, and suggests how it could be improved in future versions.

### **9.1 REVIEW OF THE AIMS**

As previously discussed, the Election System meets all of the aims originally specified for this project, and the two components of the system – Administration and Client – have been implemented. The majority of requirements have also been met. The project has researched the field of Election System in depth, and has utilized the findings from this research to ensure the system is successful. Other Election Systems have been investigated to ensure the system avoids any problems previously encountered in these systems, yet also learns from their successes

The usability and functionality of the system have been proven by conducting real-world testing, that has been carried out with forty five participants. The worksheets completed during real-world testing show that participants enjoyed their experience with our system, and agree that there is a need for this type of system for the General Elections.

### **9.2 FUTURE ENHANCEMENT**

This section examines some ways in which this system could be extended in the future.

#### **9.2.1 Suggested Revision to Design and Implementation**

Possible areas for functionality and usability improvements have been discussed throughout the previous chapter. The main areas can be summarized into the following points.



### **9.2.1.1 Touch Screen Facility**

Instead of allowing voters to only use the System Application for vote casting we could also provide a Touch Screen facility for the ease of users. There by facilitating the voters to cast vote by simply touching the desired candidate belonging to particular party or will be notified if they have already cast the vote.

### **9.2.1.2 NIC Card Reader Facility**

Our system could also be used to facilitate the voters by providing the NIC Card reader provision through which the voters just have to swap their NIC Cards in the Card Reader. So, the voter's NIC# is captured by the application which will be useful for the further vote casting procedure.

### **9.2.1.3 Screen Resolution Management**

Presently the application is running on the monitor resolution setting of 1024\*768, but for future consideration we will enhance this to support for 800\*600 or all previous resolution settings.

### **9.2.1.4 Diverse Network Support**

In future considerations we will provide our application with the capability to support several Network configurations. Our application is currently running on a LAN/WAN environment. It is desired that the system will be capable to switch to another network configuration if its current configuration failed.

### **9.2.1.5 Endorsement for Local Bodies Elections**

The Election System could be molded to fulfill the Union Counsel election process, by slightly changing the structure of existing system. Relevant changes according to the structure of the Union Council will be adapted to wider the scope of the System.



#### **9.2.1.6 Recognition Support**

In order to enhance the security aspect of the existing System it is suggested to implement facial and Iris recognition it will also support to handle the person who are disable.

### **9.3 SUMMARY**

This chapter has concluded that the original aims of this project have been met, and future developments and possibilities for further work have been identified. The project as a whole can be considered a success.



## APPENDIX A

### REPLICATION

Replication enables data and database objects to be copied and modified from one database to another across different networks and platforms. Yet, the process of synchronization maintains the consistency of the database. The physical separation of the databases and latency are the integral part of the design process in replication. These characteristics enhance, among other things, the performance of the application. Other benefits, as cited by Books Online, but not limited to, includes facilitating greater autonomy to users who can work with a local copy of the database and then transfer the changes to remote or mobile users across the network or over the Internet.

MSSQLServer 2000 permits 3 different kinds of replication. They are snapshot, transactional and merge. Snapshot makes a copy of the data and propagates the changes of the whole set of data rather than individual transactions, thereby making it a discontinuous process and entailing a higher degree of latency. Transaction replication allows incremental changes of data to be transferred either continuously or at specific time intervals. Merge replication permits a higher degree of autonomy and allows the subscribers to update changes and then propagates the changes to the publishers which in turn transfers to other subscribers.

#### Replication Architecture

Replication is a set of technologies that allows you to keep copies of the same data on multiple sites, sometimes covering hundreds of sites.

Replication uses a publish-subscribe model for distributing data:

- A Publisher is a server that is the source of data to be replicated. The Publisher defines an article for each table or other database object to be used as a replication source. One or more related articles from the same database are organized into a publication. Publications are convenient ways to group related data and objects that you want to replicate together.





- A Subscriber is a server that receives the data replicated by the publisher. The Subscriber defines a subscription to a particular publication. The subscription specifies when the Subscriber receives the publication from the Publisher, and maps the articles to tables and other database objects in the Subscriber.
- A Distributor is a server that performs various tasks when moving articles from Publishers to Subscribers. The actual tasks performed depend on the type of replication performed.

Microsoft® SQL Server™ 2000 also supports replication to and from heterogeneous data sources. OLE DB or ODBC data sources can subscribe to SQL Server publications. SQL Server can also receive data replicated from a number of data sources, including Microsoft Exchange, Microsoft Access, Oracle, and DB2.

### **Replication Types**

SQL Server 2000 uses three types of replication:

- **Snapshot replication**

Snapshot replication copies data or database objects exactly as they exist at any moment. Snapshot publications are typically defined to happen on a scheduled basis. The Subscribers contain copies of the published articles as they existed at the last snapshot. Snapshot replication is used where the source data is relatively static, the Subscribers can be slightly out of date, and the amount of data to replicate is small.

- **Transactional replication**

In transactional replication, the Subscribers are first synchronized with the Publisher, typically using a snapshot, and then, as the publication data is modified, the transactions are captured and sent to the Subscribers. Transactional integrity is maintained across the Subscribers by having all modifications be made at the Publisher, and then replicated to the Subscribers. Transactional replication is used when data must be replicated as it is modified, you must preserve the transactions, and the Publishers and Subscribers are reliably and/or frequently connected through the network.



- **Merge replication**

Merge replication lets multiple sites work autonomously with a set of Subscribers, and then later merge the combined work back to the Publisher. The Subscribers and Publisher are synchronized with a snapshot. Changes are tracked on both the Subscribers and Publishers. At some later point, the changes are merged to form a single version of the data. During the merge, some conflicts may be found where multiple Subscribers modified the same data. Merge replication supports the definition of conflict revolvers, which are sets of rules that define how to resolve such conflicts. Custom conflict resolver scripts can be written to handle any logic that may be needed to resolve complex conflict scenarios properly. Merge replication is used when it is important for the Subscriber computers to operate autonomously (such as a mobile disconnected user), or when multiple Subscribers must update the same data.



## APPENDIX B

### FINGERPRINT IDENTIFICATION: A *Literature Review*

The quality of the ridge structures in a fingerprint image is an important characteristic, as the ridges carry the information of characteristic features required for minutiae extraction. Ideally, in a well-defined fingerprint image, the ridges and valleys should alternate and flow in locally constant direction. This regularity facilitates the detection of ridges and consequently, allows minutiae to be precisely extracted from the thinned ridges. However, in practice, a fingerprint image may not always be well defined due to elements of noise that corrupt the clarity of the ridge structures. This corruption may occur due to variations in skin and impression conditions such as scars, humidity, dirt, and non-uniform contact with the fingerprint capture device. Thus, image enhancement techniques are often employed to reduce the noise and enhance the definition of ridges against valleys. This chapter provides discussion on the methodology and implementation of a fingerprint image enhancement algorithm. First, I will review existing techniques in the field of fingerprint image enhancement. This will be followed by a description of the methods used to implement the enhancement algorithm. The results of the experiments involving each stage of the fingerprint enhancement algorithm will then be presented and discussed.

One of the most widely cited fingerprint enhancement techniques is the method employed by Hong, which is based on the convolution of the image with Gabor filters tuned to the local ridge orientation and ridge frequency. The main stages of this algorithm include normalization, ridge orientation estimation, ridge frequency estimation and filtering. The first step in this approach involves the normalization of the fingerprint image so that it has a prespecified mean and variance. Due to imperfections in the fingerprint image capture process such as non-uniform ink intensity or non-uniform contact with the fingerprint capture device, a fingerprint image may exhibit distorted levels of variation in grey-level values along the ridges and valleys. Thus, normalization is used to reduce the effect of these variations, which facilitates the subsequent image enhancement steps.



An orientation image is then calculated, which is a matrix of direction vectors representing the ridge orientation at each location in the image. The widely employed gradient-based approach is used to calculate the gradient, which makes use of the fact that the orientation vector is orthogonal to the gradient. Firstly, the image is partitioned into square blocks and the gradient is calculated for every pixel, in the  $x$  and  $y$  directions. The orientation vector for each block can then be derived by performing an averaging operation on all the vectors orthogonal to the gradient pixels in the block. Due to the presence of noise and corrupted elements in the image, the ridge orientation may not always be correctly determined. Given that the ridge orientation varies slowly in a local neighbourhood, the orientation image is then smoothed using a low-pass filter to reduce the effect of outliers. The next step in the image enhancement process is the estimation of the ridge. The frequency image defines the local frequency of the ridges contained in the fingerprint. Firstly, the image is divided into square blocks and an oriented window is calculated for each block. For each block, an x-signature signal is constructed using the ridges and valleys in the oriented window. The signature is the projection of all the grey level values in the oriented window along a direction orthogonal to the ridge orientation. Consequently, the projection forms a sinusoidal-shape wave in which the centre of a ridge maps itself as a local minimum in the projected wave. The distance between consecutive peaks in the x-signature can then be used to estimate the frequency of the ridges.

Fingerprint enhancement methods based on the Gabor filter have been widely used to facilitate various fingerprint applications such as fingerprint matching and fingerprint classification. Gabor filters are bandpass filters that have both frequency-selective and orientation-selective properties, which means the filters can be effectively tuned to specific frequency and orientation values. One useful characteristic of fingerprints is that they are known to have well defined local ridge orientation and ridge frequency. Therefore, the enhancement algorithm takes advantage of this regularity of spatial structure by applying Gabor filters that are tuned to match the local ridge orientation and frequency.

Based on the local orientation and ridge frequency around each pixel, the Gabor filter is applied to each pixel location in the image. The effect is that the filter enhances the ridges



oriented in the direction of the local orientation, and decreases anything oriented differently. Hence, the filter increases the contrast between the foreground ridges and the background, whilst effectively reducing noise.

An alternative approach to enhancing the features in a fingerprint image is the technique employed by Sherlock called directional Fourier filtering. The previous approach was a spatial domain technique that involves spatial convolution of the image with filters, which can be computationally expensive. Alternatively, operating in the frequency domain allows one to efficiently convolve the fingerprint image with filters of full image size.

The image enhancement process begins by firstly computing the orientation image. In contrast to the previous method, which estimates the ridge orientation using a continuous range of directions, this method uses a set of only 16 directions to calculate the orientation. An image window is centered at a point in the raw image, which is used to obtain a projection of the local ridge information. The image window is then rotated in each of the 16 equally spaced directions, and in each direction a projection along the window's y axis is formed. The projection with the maximum variance is used as the dominant orientation for that point in the image. This process is then repeated for each pixel to form the orientation image.

Similar to the filtering stage applied by Hong after the orientation image has been computed, the raw image is then filtered using a set of band pass filters tuned to match the ridge orientation. The image is firstly converted from the spatial domain into the frequency domain by application of the two-dimensional discrete Fourier transform. The Fourier image is then filtered using a set of 16 Butterworth filters with each filter tuned to a particular orientation. The number of directional filters corresponds to the set of directions used to calculate the orientation image. After each directional filter has been independently applied to the Fourier image, the inverse Fourier transform is used to convert each image back to the spatial domain, thereby producing a set of directionally filtered images called pre-filtered images.

The next step in the enhancement process is to construct the final filtered image using the pixel values from the pre-filtered images. This requires the value of the ridge orientation at each pixel in the raw image and the filtering direction of each pre-filtered image. Each



point in the final image is then computed by selecting, from the pre-filtered images the pixel value whose filtering direction most closely matches the actual ridge orientation. The output of the filtering stage is an enhanced version of the image that has been smoothed in the direction of the ridges.

Lastly, local adaptive thresholding is applied to the directionally filtered image, which produces the final enhanced binary image. This involves calculating the average of the grey-level values within an image window at each pixel, and if the average is greater than the threshold, then the pixel value is set to a binary value of one; otherwise, it is set to zero. The grey-level image is converted to a binary image, as there are only two levels of interest, the foreground ridges and the background valleys.

Overall, it can be seen that most techniques for fingerprint image enhancement are based on filters that are tuned according to the local characteristics of fingerprint images. Both of the examined techniques employ the ridge orientation information for tuning of the filter. However, only the approach by Hong. takes into account the ridge frequency information, as Sherlock's approach assumes the ridge frequency to be constant. By using both the orientation and ridge frequency information, it allows for accurate tuning of the Gabor filter parameters, which consequently leads to better enhancement results. Hence, I have chosen to employ the Gabor filtering approach by Hong et al. to perform fingerprint image enhancement.

## Methodology

This section describes the methods for constructing a series of image enhancement techniques for fingerprint images. The algorithm we have implemented is built on the techniques developed by Hong. This algorithm consists of four main stages:

- normalization,
- orientation estimation,
- ridge frequency estimation, and
- Gabor filtering.



In addition to these four stages, we have implemented three additional stages that include:

- segmentation,
- binarisation, and
- Thinning.

In this section, we will discuss the methodology for each stage of the enhancement algorithm, including any modifications that have been made to the original techniques.

## Segmentation

The first step of the fingerprint enhancement algorithm is image segmentation. Segmentation is the process of separating the foreground regions in the image from the background regions. The foreground regions correspond to the clear fingerprint area containing the ridges and valleys, which is the area of interest. The background corresponds to the regions outside the borders of the fingerprint area, which do not contain any valid fingerprint information. When minutiae extraction algorithms are applied to the background regions of an image, it results in the extraction of noisy and false minutiae. Thus, segmentation is employed to discard these background regions, which facilitates the reliable extraction of minutiae.

In a fingerprint image, the background regions generally exhibit a very low grey-scale variance value, whereas the foreground regions have a very high variance. Hence, a method based on variance thresholding can be used to perform the segmentation. Firstly, the image is divided into blocks and the grey-scale variance is calculated for each block in the image. If the variance is less than the global threshold, then the block is assigned to be a background region; otherwise, it is assigned to be part of the foreground.



## Normalisation

The next step in the fingerprint enhancement process is image normalisation. Normalisation is used to standardise the intensity values in an image by adjusting the range of grey-level values so that it lies within a desired range of values. Let  $I(i; j)$  represent the grey-level value at pixel  $(i; j)$ , and  $N(i; j)$  represent the normalised grey-level value at pixel  $(i; j)$ .

## Orientation estimation

The orientation field of a fingerprint image defines the local orientation of the ridges contained in the fingerprint. The orientation estimation is a fundamental step in the enhancement process as the subsequent Gabor filtering stage relies on the local orientation in order to effectively enhance the fingerprint image. The least mean square estimation method employed by Hong. is used to compute the orientation image. However, instead of estimating the orientation block-wise, We have chosen to extend their method into a pixel-wise scheme, which produces a finer and more accurate estimation of the orientation field.

## Gabor filtering

Once the ridge orientation and ridge frequency information has been determined, these parameters are used to construct the even-symmetric Gabor filter. A twodimensional Gabor filter consists of a sinusoidal plane wave of a particular orientation and frequency, modulated by a Gaussian envelope. Gabor filters are employed because they have frequency-selective and orientation-selective properties.

These properties allow the filter to be tuned to give maximal response to ridges at a specific orientation and frequency in the fingerprint image. Therefore, a properly tuned Gabor filter can be used to effectively preserve the ridge structures while reducing noise.





## Thinning

The final image enhancement step typically performed prior to minutiae extraction is thinning. Thinning is a morphological operation that successively erodes away the foreground pixels until they are one pixel wide. A standard thinning algorithm is employed, which performs the thinning operation using two subiterations.

This algorithm is accessible in JAVA via the 'thin' operation under the user-defined function. Each subiteration begins by examining the neighbourhood of each pixel in the binary image, and based on a particular set of pixel-deletion criteria, it checks whether the pixel can be deleted or not. These subiterations continue until no more pixels can be deleted.

The application of the thinning algorithm to a fingerprint image preserves the connectivity of the ridge structures while forming a skeletonised version of the binary image. This skeleton image is then used in the subsequent extraction of minutiae.

## Fingerprints Architecture

Most automatic systems for fingerprints comparison are based on the minutiae matching. Minutiae are local discontinuities in the fingerprint pattern. A total of 150 different minutiae types have been identified. In practice only ridge ending and ridge bifurcation minutiae types are used in the fingerprint recognition. Examples of minutiae are shown in figure.

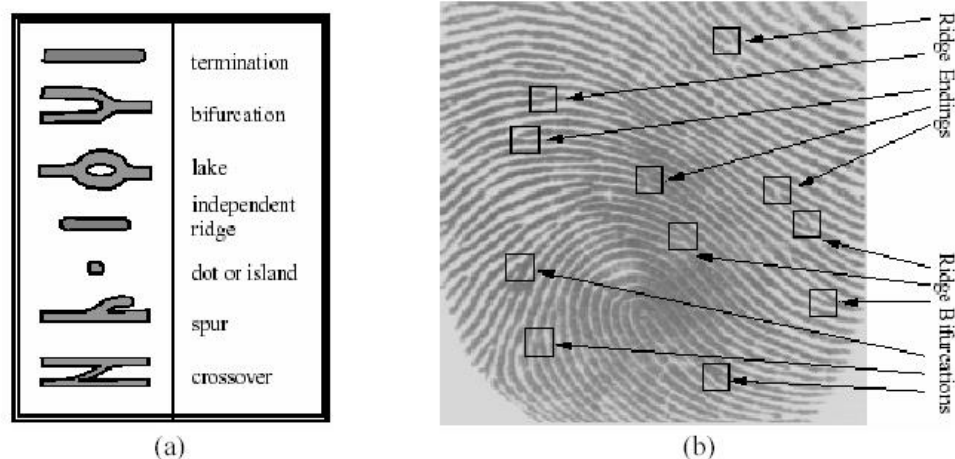
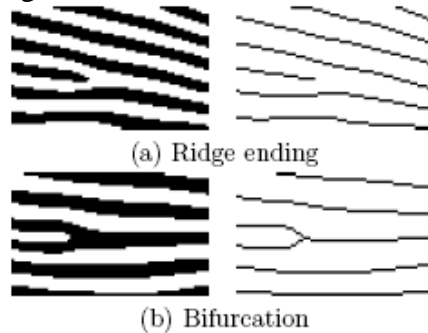


Figure 1. (a) Different minutiae types, (b) Ridge ending & Bifurcation



Ridge endings are the points where the ridge curve terminates, and bifurcations are where a ridge splits from a single path to two paths at a Y-junction.

A ridge ending occurs when the ridge flow abruptly terminates and a ridge bifurcation is marked by a fork in the ridge flow.

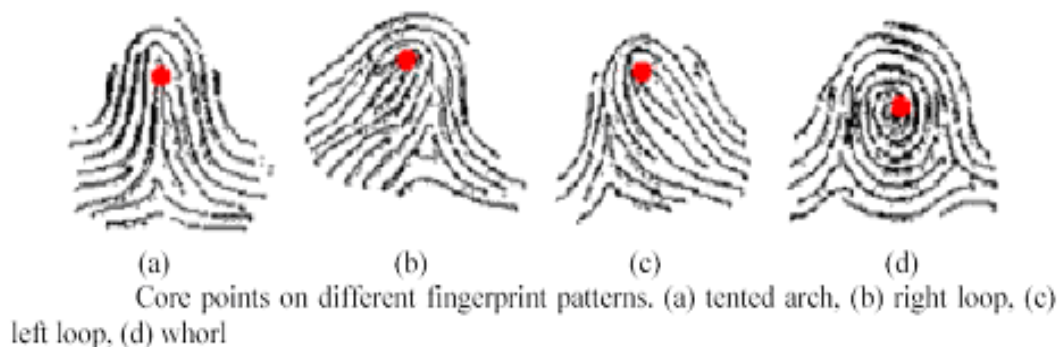


The fingerprint of an individual is unique and remains unchanged over a lifetime. A fingerprint is formed from an impression of the pattern of ridges on a finger. A ridge is defined as a single curved segment, and a valley is the region between two adjacent ridges. The minutiae, which are the local discontinuities in the ridge flow pattern, provide the features that are used for identification.

## Fingerprint Classification

Usually, the center or reference point of the fingerprint image is what is called the “core” point. A core point, is located at the approximate center, is defined as the topmost point on the innermost upwardly curving ridgeline.

The human fingerprint is comprised of various types of ridge patterns, traditionally classified according to the decades-old Henry system: left loop, right loop, arch, whorl, and tented arch. Loops make up nearly  $\frac{2}{3}$  of all fingerprints, whorls are nearly  $\frac{1}{3}$ , and perhaps 5-10% is arches. Figure 2 shows some fingerprint patterns with the core point marked.



## Approaches

A (three-class) categorization of fingerprint matching approaches is:

- ***Correlation-based matching***: two fingerprint images are superimposed and the correlation (at the intensity level) between corresponding pixels is computed for different alignments (e.g., various displacements and rotations).
- ***Minutiae-based matching***: minutiae are extracted from the two fingerprints and stored as sets of points in the two-dimensional plane. Minutiae matching essentially consist of finding the alignment between the template and the input minutiae sets that results in the maximum number of minutiae pairings.
- ***Ridge feature-based matching***: minutiae extraction is difficult in very low-quality fingerprint images, whereas other features of the fingerprint ridge pattern (e.g., local orientation and frequency, ridge shape, texture information) may be extracted more reliably than minutiae, even though their distinctiveness is generally lower. The approaches belonging to this family compare fingerprints in term of features extracted from the ridge pattern.



## **Fingerprint Verification SDK**



**Authentication Software  
Development Kit**

### **User Manual**

### **SDK Components**

### **Imaging Suite**

First, Image Capture calibrates the fingerprint sensor, chooses default sensor settings, and performs sensor diagnostics. Then, Image Capture requests up to 90,000 bytes of raw image data from the sensor, and stores the data in an image array.

### **Automatic Gain Control with DFX**

Image Plus and Image Quick enhance Image Capture's functions by providing feedback on the quality of the captured image and automatically adjusting the sensitivity of the sensor to create the best image for wet, dry or difficult-to-read fingers. They perform Automatic Gain Control (AGC) on a column-by-column basis to obtain images of uniformly high contrast and low noise. In addition to AGC, Image Plus invokes DFX (Difficult Finger Extraction). DFX eliminates background noise and adjusts gain to produce clean, high-quality images for both dry and excessively wet fingers.



## Verification Suite™

Verification Suite's modules then process the image:

- Image Quality analyzes the quality of the captured image. Error codes flag problems such as an incorrect image position, no finger present on the sensor, or fingers that are too wet or too dry.
- Image Enhancement reduces noise, enhances ridges in the captured grayscale image, and converts the image to binary form.
- Minutiae Extraction identifies the ridge endings and bifurcations, called minutiae, then extracts and stores a unique, 300-byte minutia data template from a fingerprint.
- Fingerprint Verification verifies an individual's identity. A minutia template from a live reading of a finger placed on the sensor is compared to the stored minutia data template. If the templates match, the individual's identity is verified. If the templates do not match, the attempt is rejected.



## Imaging Suite

### VFPSselectSensor( )

◆ ◆ Image Plus ◆ ◆ Image Capture

---

**NOTE:** This function must be called prior to `VFPSinit( )` or else the default device type will be opened.

---

`VFPSselectSensor( )` allows an application—at run-time—to choose between using the 5<sup>th</sup>Sense USB or Parallel module, the 5<sup>th</sup>Sense USB or Parallel Combo module, or the embedded sensor, regardless of which library the application is linked to.

`VFPSselectSensor( )` has an override parameter that allows you to change the device type to load (1) or to select one if none is loaded (0). Since `VFPSselectSensor( )` always returns the loaded device type, setting override to 0 lets the user find out which type of device is loaded without actually changing the device.

---

**USAGE NOTE:** To switch between devices on the fly, `VFPSclose( )` must be called to unload the current device before `VFPSselectSensor( )` can be called to load it. Of course, a call to `VFPSinit( )` is required after every sensor type change.

---

#### Function Prototype

```
long VFPSselectSensor (long sensorType, long override)
```

### VFPSinit( )

◆ ◆ Image Plus ◆ ◆ Image Capture

`VFPSinit( )` initializes the sensor to its default values and opens the device. `VFPSinit( )` must be used before any other calls.

The functionality of `VFPSinit( )` is determined by its flags. For example, you can set `VFPSinit( )` flags to calculate optimum values for the sensor to compensate for chip variations, perform DFX calibration, and so forth.

#### Function Prototype

```
SensorHandle_t VFPSinit (long sensorID, long *numRows, long
*numColumns, long *mode, char *configFile, long flags,
unsigned char *usrBkgndImage, void (*callbackfcn)(long)).
```



## VFPSInitialize ()

### ◆ ◆ Image Plus ◆ ◆ Image Capture

`VFPSInitialize ()` is a subset of `VFPSinit ()`, and provides backward compatibility with earlier product releases. The `VFPSInitialize ()` function will not be available in future product versions.

## VFPSgetImage()

### ◆ ◆ Image Plus ◆ ◆ Image Capture

`VFPSgetImage ()` is the basic call to get an image. Using Veridicom proprietary algorithm, `VFPSgetImage ()` returns a much improved image over the raw image obtained from the sensor. It reduces the effects of any row or column that `VFPSInitialize ()` finds nonfunctional.

For improved image quality, enable Image Plus DFX by setting `VFPS_IMAGEPLUS` flag to `VFPSgetImage ()`.

`VFPSgetImage ()` grabs an image from the sensor into the array `imageData`. The sensor must be initialized using `VFPSinit ()` before this call, and must be closed using `VFPSclose ()` after this call. Whereas initializing and closing must each be done once in the application, `VFPSgetImage ()` can be done any number of times.

### Function Prototype

```
int VFPSgetImage (SensorHandle_t handle, long mode, unsigned
char *imageData, long flags)
```



## Verification Suite

### fpProcess()

The program accepts an 8 bit grayscale fingerprint image as input and returns a list of minutiae. If `imageOut` is a valid, allocated memory pointer, then the processed binary image will be stored in the memory pointed to. If `imageOut` is a NULL pointer, the processed binary image will not be written. This image has the same size as the initial image and uses the same amount of storage. If `dFlag` is set (not zero), then the minutia locations are indicated upon the output image by box outlines. The lighter-framed boxes indicate endpoint minutiae, and the darker-framed boxes indicate bifurcation minutiae. The short vector emanating from a box gives an approximate indication of the direction of the minutia: from endpoint into the ridge for endpoints, and from single ridge toward two ridges for a bifurcation.

#### Function Prototype

```
long fpProcess (unsigned char HUGEAPR *imageIn, long height,
long width, void *minutiae, long *nMinutiae, unsigned char
HUGEAPR *imageOut, long dFlag, long reserved);
```

### matchprints()

This function matches two fingerprint templates as represented by the data in `searchMinu` and `fileMinu`. Match scores indicate the number of minutiae neighborhoods found to exist in both compared fingerprints, and the score ranges from 3 to 25. CFMEF template is used for the function.

#### Function Prototype

```
long matchprints (void *searchMinu, void *fileMinu, long
nSecurityLevel);
```





# Fingerprint Enrollment

---

Fingerprint Enrollment is to register the fingerprint template for later recognition. A good enrollment is crucial for a reliable fingerprint recognition system. Here are some general factors that may influence the enrollment:

- **Finger Position.** Place the center of finger in the sensor center. The fingerprint may be positioned far left, far right, far up or far down. For instance, if a user enrolled a far left fingerprint. During verification, the user may press far right. Since there is no overlap between enrollment and verification templates, the user may be rejected. If the user enrolls the finger in the middle, there is a better chance of being accepted.
- **Finger area.** The best advice is to cover the sensor area completely with finger to ensure the maximum fingerprint surface contact area. A common mistake is to touch the sensor with the tip of the finger, which contains too few minutiae.
- **Finger rotation.** It will be good for the user to keep the fingerprint rotation minimal during the enrollment. The rotation should be within  $\pm 45$  degrees during enrollment.
- **Finger condition, very dry or wet finger.** Although Veridicom sensor is designed to handle dry or wet conditions, it is wise to handle specially of very dry or wet fingers. The user can wipe wet fingers with cloth or paper towel. For dry finger, the user can moisturize the finger by breathing on it, or by touching the forehead to pick up surface oil. The image quality could be improved tremendously by taking care of the dry/wet condition during enrollment.
- **Finger pressure.** Use medium pressure. Excessive pressure may distort the image and adhere ridges together. Too little pressure would lead to a small fingerprint area or dry fingerprint.



## REFERENCES

1. Study of Pakistan Election System as “Intelligent e-Election” Muhammad Nadeem, Dr. Javaid R. Laghari, SZABIST KARACHI.
2. Election Commission of Pakistan Website “<http://www.ecp.gov.pk/>”
3. “Electronic Voting – Evaluating the Threat,” Michael Ian Shamos, International Conference on Computers, Freedom, and Privacy, Burlingame, California, 1993.
4. UniVote Interactive Public Voting System by Nicholas James “<http://www.lancs.ac.uk/ug/dayn>”
5. Electronic Voting: Algorithmic and Implementation Issues Robert Kofler, Robert Krimmer, Department Production Management, Vienna University for Business Administration and Economics
6. Veridicom, Inc. Information can be found at home page of the Veridicom on the World Wide Web at “<http://www.veridicom.com>”.

