

**Customize and encrypted socket connection for polling application (Usecase : Online voting management)**

**High Level Design & Low Level Design**

The purpose of this document is to provide  a template for documenting both HLD & LLD. 

**Document Control :**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Revision History** | | | | | | | |
|  |  |  |  |  |  |  |  |
| **Date** | **Version** | **Author** | **Brief Description of Changes** | | | **Approver Signature** | |
| 16/11/2022 | Version 1.0 |  |  | | |  | |
|  |  |  |  | | |  | |
|  |  |  |  | | |  | |
|  |  |  |  | | |  | |
|  |  |  |  | | |  | |

**Contents**

[**1. Introduction**](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.3znysh7) **5**

[1.1. Intended Audience](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.2et92p0) 5

[1.2. Acronyms/Abbreviations](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.tyjcwt) 5

[1.3. Project Purpose](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.3dy6vkm) 5

[1.4. Key Project Objectives](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.1t3h5sf) 5

[1.5. Project Scope and Limitation](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.4d34og8) 5

[*1.5.1. In Scope*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.2s8eyo1) 6

[1.6. Functional Overview](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.3rdcrjn) 6

[1.7. Assumptions, Dependencies & Constraints](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.26in1rg) 6

[1.8. Risks](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.lnxbz9) 6

[**2. Design Overview**](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.35nkun2) **7**

2[.1 Design Objectives](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.1ksv4uv) 8

[*2.1.1. Recommended Architecture*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.44sinio) 8

[2.2. Architectural Strategies](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.2jxsxqh) 8

[*2.2.1. Design Alternative*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.z337ya) 8

[*2.2.2. Reuse of Existing Common Services/Utilities*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.3j2qqm3) 8

[*2.2.3. Creation of New Common Services/Utilities*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.1y810tw) 9

[*2.2.4. User Interface Paradigms*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.4i7ojhp) 9

[*2.2.5. System Interface Paradigms*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.2xcytpi) 9

[*2.2.6. Error Detection / Exceptional Handling*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.1ci93xb) 9

[*2.2.7. Memory Management*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.3whwml4) 9

[*2.2.8. Performance*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.2bn6wsx) 9

[*2.2.9. Security*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.qsh70q) 9

[*2.2.10. Concurrency and Synchronization*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.3as4poj) 9

[**3. System Architecture**](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.49x2ik5) **10**

[3.1. System Architecture Diagram. (Not Necessary)](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.2p2csry) 11

[3.2. System Use-Cases](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.147n2zr) 12

[3.3. Subsystem Architecture](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.3o7alnk) 13

[3.4. System Interfaces](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.23ckvvd) 14

[*3.4.1. Internal Interfaces*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.ihv636) 14

[*3.4.2. External Interfaces*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.32hioqz) 14

[**4. Detailed System Design**](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.1hmsyys) **14**

[4.1. Key Entities](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.41mghml) 14

[4.2. Detailed-Level Database Design](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.2grqrue) 14

[*4.2.1. Data Mapping Information 1*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.vx1227)4

[4.3. Archival and retention requirements](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.1v1yuxt) 14

[4.4. Disaster and Failure Recovery](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.4f1mdlm) 15

[4.5. Business Process workflow](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.2u6wntf) 15

[4.6. Business Process Modeling and Management (as applicable)](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.19c6y18) 15

[4.7. Business Logic](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.3tbugp1) 15

[4.8. Variables](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.28h4qwu) 15

[4.9. Activity / Class Diagrams (as applicable)](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.nmf14n) 15

[4.10. Data Migration](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.37m2jsg) 15

[*4.10.1. Architectural Representation*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.1mrcu09) 15

[*4.10.2. Architectural Goals and Constraints*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.46r0co2) 15

[*4.10.3. Logical View*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.2lwamvv) 15

[*4.10.4. Architecturally Significant Design Packages*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.111kx3o) 15

[*4.10.5. Data model*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.3l18frh) 15

[*4.10.6. Deployment View*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.1egqt2p) 16

[**5. Environment Description**](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.3ygebqi) **16**

[5.1. Time Zone Support](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.2dlolyb) 16

[5.2. Language Support](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.sqyw64) 16

[5.3. User Desktop Requirements](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.3cqmetx) 16

[5.4. Server-Side Requirements](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.1rvwp1q) 16

[*5.4.1. Deployment Considerations*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.4bvk7pj) 16

[*5.4.2. Application Server Disk Space*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.2r0uhxc) 16

[*5.4.3. Database Server Disk Space*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.1664s55) 17

[*5.4.4. Integration Requirements*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.3q5sasy) 17

[5.5. Configuration](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.1jlao46) 17

[*5.5.1. Operating System*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.43ky6rz) 17

[*5.5.2. Database*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.2iq8gzs) 17

[*5.5.3. Network*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.xvir7l) 17

[*5.5.4. Desktop*](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.3hv69ve) 17

[**6. References**](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.1x0gk37) **17**

[**7. Appendix**](https://docs.google.com/document/d/12Zkn-7XhoUFHzHxoF7Trfcw706LFqpnO/edit#heading=h.4h042r0) **18**

**1. Introduction**

Online Voting is a web-based voting system that will help you manage your elections easily and securely. This voting system can be used for casting votes during the elections etc. In this system the voter do not have to go to the polling booth to cast their vote. They can use their personal computer to cast their votes. There is a database which is maintained in which all the name of the voters with their Voter ID is stored.

OVS also known as e-voting is a term encompassing several different types of voting embracing both electronic means of counting votes. OV is an electronic way of choosing leaders via a web driven. The advantage of OV over the common queue method is that the voters have the choice of voting at their own free time and there is reduce congestion. It also minimizes on errors of vote counting. The individual votes are submitted in a database which can be queried to find out who of the aspirants for a given post has the highest number of votes.

This system is geared toward increasing the voting percentage since it has been noted that with the old voting method (Queue system), the voter turnout has been a wanting case. with system in place also, if high security applied, cases of false votes shall be reduced.

The crucial points that this emphasizes on are listed below :

* Require less number of staff during the elections
* This system is a lot easier to independently moderate the elections and subsequently reinforce its transparency and fairness
* Less capital, less effort and less labor Intensive, as the primary cost and effort will focus primarily on creating, managing, and running a secure online portal.
* Increased number of voters as individual will find it easier and more convenient to vote, especially those abroad.

**1.1   Intended Audience**

|  |  |
| --- | --- |
| BU Authority |  |
|  |  |

**1.2  Acronyms/Abbreviations**

|  |  |
| --- | --- |
| OVS | Online voting system |
| OV | Online voting |
| TCP | Transmission control protocol |

**1.3  Project Purpose**

In “ONLINE VOTING SYSTEM” a voter can use his/her voting right online without any difficulty. Online Voting System is a system which enables all citizens to cast their vote online.

The purpose is to increase the voting percentage across the country, as in the present system people have to visit the booth to cast their vote and those people who live out of their home town are not able to cast vote during the elections. So due to this the voting percentage across the country is very less. Through this software those people who live out of their home town will also be able to cast their votes as this system is online.

**1.4 Key Project Objectives**

Implementing an Online voting system

**1.5  Project Scope and Limitation**

The scope of the OVS include:

-> To provide an improved voting service to the voters through fast, timely and convenient voting.

-> Reduction of the expenses incurred by the election commission during voting period, paying clerks who are assigned to ensure the pleasant and safe execution of the manual voting process.

**1.5.1 In Scope**

* Admin/User should provide a valid Credentials
* When the client is sending, the server should respond to it.
  1. **Functional Overview**
* **Admin Login** : In admin entering the login credentials with username and password which requires validation to check voters list and candidate list.
* **User Login** : In user entering the login credentials with username and password which requires validation to cast their vote.
* **Voters list :** Creating a text file for voters list who are eligible to their vote with name and voter ID
* **Client :** In the client terminal the client will send the data to server which contains the information of name of voter and voter ID. It contains the functions of sockfd() , connect() and send() for creation of socket , connecting client to server and sending the data to server.
* **Server :** In the server terminal the server checks the list of voters and the voter who is going to vote if it matches they will allow to select the candidate symbol and can cast their vote, Otherwise they can exit. One who already voted is not able to vote again.

**1.7  Assumptions, Dependencies & Constraints**

OPERATING SYSTEMS:

Operating environment for implementing TCP socktet are:

* Client/server system
* Operating system: Linux
* Platform: Ubuntu/C++
  1. **Risks**

            No Risk(As it is for educational purpose)

**2. Design Overview**

**1. Start :** This is the start block which indicates the start of the program, which will accept the admin and user credentials (like the username and password). On validation of these credentials the system will allow the admin and user to further communicate else if the credentials are wrong it will display an error message indicating “Invalid Credentials”.

**2. Login for Admin and User :** Entering the login credentials for admin to get access, similarly user is having the login credentials to cast their vote.

**3. List of voters :** Creating a text file for voters list who are eligible to cast vote with name and voter ID.

4.**Client-Server communication :** Using TCP Socket a client server communication is established.

* **At client :**
* using the function of sockfd() socket creation will be done
* using the function of connect(), it connects the client socket to server socket
* using send(), the data at client side will be send to server side
* **At server :**
* Socket creation will be done using sockfd()
* Using bind(),it binds the socket to the address and port number
* using listen(),it waits for the client to approach the server to make a connection
* using accept(),it accepts the incoming connection from client
* using receive(),it receives the data from client

1. **Checking the voter details :** At server side it will check the entered voter details matched with the voters list , they will be able to select the candidate symbol to cast their vote else they are not able to vote.
2. **Selecting the candidate symbol:** The voter is going to be select the symbol for respective candidate to cast their vote.
3. One who already casted their vote,they are not able to vote again and they will be exit.
4. **Result Declaration :** After completing the voting time, the winner will be declared according to votes allocated to each candidate.
5. **Exit :** The process will be terminated.

**2.1 Design Objectives**

* Create login credential page for Admin and user
* Implement a client-server communication using TCP socket
* Server will check whether voter name and ID exist or not.
* If exists, server will allow the voter to select the candidate symbol from the list.
* else they are going to exit from the page
* Result will be declared among the candidates who gets highest number of votes.

**2.1.1 Recommended Architecture**

Generic

**2.2 Architectural Strategies**

* Header files
* Macros

**2.2.1  Design Alternative**

NA

**2.2.2 Reuse of Existing Common Services/Utilities**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<arpa/inet.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<netdb.h>

#include<regex.h>

#include<unistd.h>

**2.2.3 Creation of New Common Services/Utilities**

           NA

**2.2.4 User Interface Paradigms**

                          Command Line Interface: Terminal

**2.2.5 System Interface Paradigms**

             Command Line Interface: Terminal

**2.2.6 Error Detection / Exceptional Handling**

             Error detection :

1. Invalid Credentials
2. Voter ID does not exist
3. Errors will be handled by perror

**2.2.7 Memory Management**

NA

**2.2.8 Performance**

NA

**2.2.9 Security**

         For security purposes the system asks for login credentials for Admin and User.

**2.2.10 Concurrency and Synchronization**

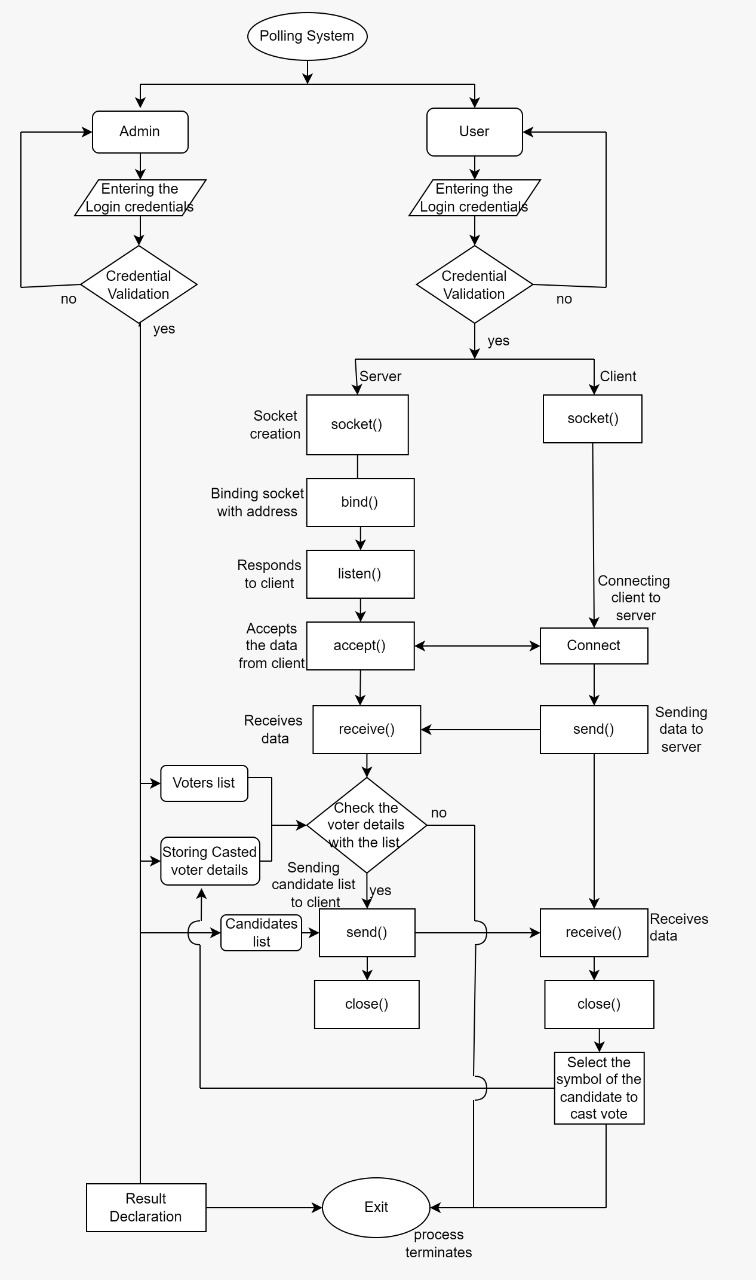
NA

**3.  System Architecture**

**LEVEL 0 DFD:**

**LEVEL 1 DFD:**

**3.1 System Architecture Diagram**

****

**3.2  System Use-Cases**

**

**3.3  Subsystem Architecture**

    NA

**3.4  System Interfaces**

      NA

**3.4.1 Internal Interfaces**

     NA

**3.4.2 External Interfaces**

     NA

1. **Detailed System Design**

The code starts by declaring the Login credentials for both admin and user individually. A text file is created for list of voters who already casted their vote with name, voter id. There is a login with admin and user. Admin login is by “Election Commision” and user login is by “voter”.

Using Socket (TCP/UDP) a client server communication is established. With this communication Voters can be able to cast their vote. One who already voted, not able to vote again. And result will be declared with the maximum number of votes.

**4.1  Key Entities**

* Valid Login Credentials
* Valid Voter name and ID

**4.2  Detailed-Level Database Design**

    NA

**4.2.1  Data Mapping Information**

             Mapping the Voter ID from server side

**4.3  Archival and retention requirements**

NA

**4.4  Disaster and Failure Recovery**

* We don’t have any control over the system. In case of failure, source code is safe.
* Use of Git.

**4.5  Business Process workflow**

    NA

**4.6  Business Process Modeling and Management (as applicable)**

                              NA

**4.7  Business Logic**

     NA

**4.8  Variables**

NA

**4.9  Activity / Class Diagrams (as applicable)**

**Pseudocode**

1)Login\_credentials

//Admin

if(Admin\_login==valid)

print Credentials are validated

else

print Credentials are invalid

exit

//User

if(User\_login==valid)

print Credentials are validated

else

print Credentials are invalid

exit

2)Client\_socket

//Socket creation

if(socket==successful)

print socket created successfully

else

print socket failed.Error!!!

//Connection between client and server

if(connect==successful)

print connection established

else

print connection failed

//Send data to server

if(send==successful)

print data sent successfully

else

print data not sent

//Receive the data from server

if(receive==successful)

print data received successfully

else

print data not received.Error!!!

3)Server\_socket

//Socket creation

if(socket==successful)

print socket created successfully

else

print socket failed.error!!!

//Binding with socket

if(bind==successful)

print binding successful

else

print binding failed.Error!!!

//Listenening to client

if(listen==successful)

print listening....

else

print listening failed

//Accept

if(accept==successsful)

print client connected with IP and port number

else

print accept failed

//Send data to server

if(send==successful)

print data sent successfully

else

print data not sent

//Receive the data from server

if(receive==successful)

print data received successfully

else

print data not received

4)//Creation of textfile for voters list

Voters list should be create with the voter name and ID

5)//Creation of textfile for Candidates list

Candidate name and party symbol name should be create in a textfile

6)//creation of textfile for storing casted voter details

Voter name and ID should be store in a textfile who done their voting

7)//User enters voter\_details

voter ID and name should be entered from client side which sends this information to server.

8)//Checking the voters list

if(voter\_details exists in voters list)

if(voter\_details not exists in casted voter list)

print Welcome voter

else

print Sorry! You have already casted a vote.You can't vote again.

else

print Sorry! Your details are not in the authorized voters list

9)//Selecting the party name

if(user selects respective candidate)

print Congratulations! you have successsfully casted your vote

else

print Error !! Invalid choice

10)//Result declaration

Counting the number of votes for each party

Declaring the winner for every vote casting with symbol

11)//Exit

Process will terminate

**4.10 Data Migration**

NA

**4.10.1 Architectural Representation**

NA

**4.10.2 Architectural Goals and Constraints**

The project is just for educational purposes.

**4.10.3 Logical View**

NA

**4.10.4 Architecturally Significant Design Packages**

NA

**4.10.5 Data model**

NA

**Legacy system data model**

**Proposed system data model**

**Interface data model**

**4.10.6 Deployment View**

NA

1. **Environment Description**

* **GCC:** In Linux, the GCC stands for GNU Compiler Collection. It is a compiler system for the various programming languages. It is mainly used to compile the C and C++ programs.
* **Socket Programming:** Socket programming is a way of connecting two nodes on a network to communicate with each other. One socket(node) listens on a particular port at an IP, while the other socket reaches out to the other to form a connection. The server forms the listener socket while the client reaches out to the server.
* **UBUNTU:** Ubuntu is an open-source operating system (OS) based on the Debian GNU/Linux distribution. Ubuntu incorporates all the features of a Unix OS with an added customizable GUI, which makes it popular in universities and research organizations. Ubuntu is primarily designed to be used on personal computers, although a server edition does also exist.
* **GITHUB:** GitHub is a code hosting platform for version control and collaboration. It lets you and others work together on projects from anywhere. This tutorial teaches you GitHub essentials like repositories, branches, commits, and pull requests.

**5.1 Time Zone Support**

                       NA

**5.2 Language Support**

                      NA

**5.3 User Desktop Requirements**

   Linux, Ubuntu

**5.4 Server-Side Requirements**

   Linux, Ubuntu

**5.4.1 Deployment Considerations**

          NA

**5.4.2 Application Server Disk Space**

          NA

**5.4.3 Database Server Disk Space**

          NA

**5.4.4 Integration Requirements**

            NA

**5.5  Configuration**

                    NA

**5.5.1 Operating System**

Linux desktop editions with 8 GB RAM- A GUI-based LINUX system must be   used

**5.5.2 Database**

      NA

**5.5.4  Desktop**

* CPU : Intel i3/i5/i7 generation 3 and later
* RAM: 4GB or greater - For optimal performance, 6GB or 8GB are recommended if you will be running multiple browser tabs and/or multiple applications at the same time
* Internal memory:476 GB SSD/HDD.

1. **References**

<https://man7.org/linux/man-pages/index.html>

[Introduction to Sockets Programming in C using TCP/IP](https://www.csd.uoc.gr/~hy556/material/tutorials/cs556-3rd-tutorial.pdf)

<https://www.ibm.com/docs/en/zos/2.2.0?topic=reference-library-functions>

1. **Appendix**

**Change Log**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **QMS Template Version Control (Maintained by QA)** | | | | | |
|  |  |  |  |  |  |
| **Date** | **Version** | **Author** | | **Description** | |
|  | 1.0 |  | | Initial Version | |
|  |  |  | |  | |
|  |  |  | |  | |
|  |  |  | |  | |