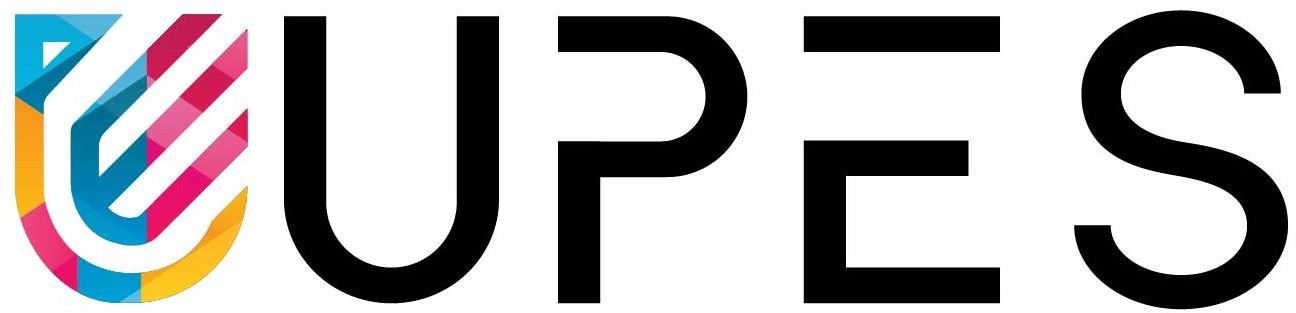
Software Requirements Specification

For

Mini Voting System Using C 30 October, 2023

Prepared by

|  |  |  |
| --- | --- | --- |
| **Specialization** | **SAP ID** | **Name** |
| DevOps | 500090924 | Muskan Singh |
| DevOps | 500090925 | Mansi Singh |
| DevOps | 500090943 | Avika Agarwal |
| DevOps | 500093014 | Shambhavee  Srivastava |



Department of Systemics School Of Computer Science

UNIVERSITY OF PETROLEUM & ENERGY STUDIES,

DEHRADUN- 248007. Uttarakhand

***INDEX***

|  |  |
| --- | --- |
| Introduction | 3 |
| Purpose of the project | 4 |
| Target Beneficiary | 4 |
| Project Scope | 5 |
| Project Description | 6 |
| Swot Analysis | 6 |
| Project Features | 7 |
| Design and implementation constraints | 8 |
| Design diagram | 10 |
| Pert Chart | 15 |
| Programming concepts | 16 |
| Flowchart | 17 |
| Technical Diagram | 20 |
| System Requirements | 21 |
| Non- functional requirements | 22 |
| References | 24 |

# Introduction

We have used the C program to implement an electronic voting system. It has following functionalities:

* 1. It can set up the new elections.
  2. It looks after the administration of the election.
  3. Banning of the user id.
  4. Keeping the voting records.
  5. And finally declaring the elections results.

Our project will include several standard C libraries and header files such as <stdio.h>,

<conio.h>, <string.h> and <stdlib.h>. All the essential functions and data types are provided by these headers.[1]

DSA: Two very well-known and important data structures that will be used in our project are: ‘typedef’ and ‘struct’.

Struct currentValidID: This will be used to represent the valid numbers of user id that are currently present which will include the year, branch code and total number of votes received. Typedef struct candidate: This will represent the information about the candidates including their IDs, Names and total number of votes that they have received.[2]

Global Variables: Our project will include several global variables also like ‘currentValidID’ structure, then an array will be used for storing the informations of the candidates (‘candidateArray’), ‘numberOfCandidates’ will be used for storing total number of candidates. And an array will be used for storing the casted votes by the students (‘studentVotes’).[3]

Utility Functions: Our project will include several utility functions like:

1. extractbatchYear: As the name suggest it will be used for extracting the year from the user id.
2. extractRollNumber.: It will be used for extracting the roll number from a user id.
3. branchCode: It will be used for matching the global branch code with the Branch code in a user id.
4. authenticateAdminID: Username and Passwords are verified in order to authenticate the administrators.
5. banIDentification: Records are maintained in a ‘Banned.txt’ file of all the user id that will be banned.
6. produceCandidates: Separate files will be maintained for each candidates.
7. discardIllegalVote: Based on the user id votes casted by the users are deleted.
8. retrieveWinner: Winner of the elections will be determined by this.
9. intiateFreshElection: New elections are initialized and users are prompted for various details of the elections.
10. aechiveElectionData: Information of the elections are saved to a file.
11. importElectionDataFromfile: Data related to elections are loaded in a file.
12. administrativeHub: Administrative Panels are provided in order to manage the elections.

Several functions will be used to check the validity of the user id and votes and also the saving votes.[4]

Student Panel: Student panel will be used by the student for casting their votes to their for casting their votes to their selected candidates by entering their user id.

Both administrative as well as student functionalities will be provided by our project will also include:

1. Error Handling
2. File Management

# Purpose of the Project

Objective of the project is to create a platform for conducting elections which will be secure, efficient and transparent. Some of the main objectives are listed below:[5]

* 1. Modernization of the voting system
  2. Enhanced Security
  3. Improved Efficiency
  4. Ensure Accessibility
  5. Reduce Administrative Burden
  6. Increase Transparency
  7. Enable Remote Voting
  8. Facilitate Data Analysis
  9. Customization
  10. Compliance
  11. Educational Purpose
  12. Research & Development
  13. Secure & Efficient Elections

# Target Beneficiary

The code you gave appears to be for an electoral programme. It is intended for election management and administration, allowing administrators to set up and manage elections, prohibit user IDs, and monitor election results. Students can use their user IDs to log in, vote, and view the list of candidates.[6]

This code's intended recipients include:

Administrators have the ability to start and administer elections, erase invalid votes, block user IDs, and examine election results.[7]

Students can log in using their user IDs to vote and examine the list of candidates.

This code provides functionality for conducting and managing elections in an academic institution or a similar setting where administrators are in charge of setting up and monitoring elections and students must vote. It enables a controlled and organised election process.[8]

# Project Scope

Problem statement of our project is creating an electronic voting machine for elections. Our project supports the following features:[9]

* 1. Admin Panel:
     + Authentication of administrator with username and password.
     + Initiation of new elections and specification of year, branch code, total number of voters and candidates.
     + Information of previous election is loaded from files.
     + Illegal votes are deleted on the basis of user IDs.
     + Specific ID of user can be banned from voting.
     + Determining and displaying result of election, which also includes the winner and number of votes that each candidate received.
  2. Student Panel:
     + Student can vote by entering their user IDs.
     + Verifying the validity of user ID on the basis of their format, year, branch code and roll number.
     + Checking if the user IDs has already voted.
     + Banned user IDs are prevented from voting.
     + List of candidates is presented for students to choose from.
     + Student are allowed to give votes to their preferred candidates.
     + For storing information of candidates, valid user IDs and votes our project defines the structures and global variables.
     + Extraction of year and roll number from user IDs is done by this project, it check if branch code matches, authenticate administrators, ban user IDs, create candidate files, delete illegal votes, determine election winner, initiate new elections, save election information in files, and load election information from files.
     + Admin Panel is present which allow administrator to perform mentioned task.
     + Student Panel is present that allow student to enter their IDs, verify their validity, check if they have already voted or are banned, vote for candidates and record their votes.[10]

# Project Description

This project is of an electronic voting system which has functionalities for both administrator and student. Following benefits is offered to society by this project:[11]

* 1. Transparency in election
  2. Efficiency
  3. Accessibility
  4. Reduced paper usage
  5. Security
  6. Data Integrity
  7. Efficient administration
  8. Result transparency
  9. Engagement
  10. Cost savings

Our project implements the following functionalities:

* + - For storing information of candidates, valid user IDs and votes our project defines the structures and global variables.
    - Extraction of year and roll number from user IDs is done by this project, it check if branch code matches, authenticate administrators, ban user IDs, create candidate files, delete illegal votes, determine election winner, initiate new elections, save election information in files, and load election information from files.
    - Admin Panel is present which allow administrator to perform mentioned task.
    - Student Panel is present that allow student to enter their IDs, verify their validity, check if they have already voted or are banned, vote for candidates and record their votes.[12]

Advantages:

1. Efficient Record Keeping: For future reference and audits, electronic systems may effectively preserve and manage the voter data, candidate data and election results. 2.Quick Result: Election results can be calculated and announced quickly with electronic voting which also reduced the waiting times.

3.Reduced Paper Usage: Electronic voting eliminates the need for paper ballots, saving money and protecting the environment.[13]

# SWOT Analysis

Strengths-

Efficiency- C is noted for its low-level system access and efficiency, which makes it ideal for resource-constrained situations. A simple voting system written in C can operate effectively on low-cost hardware.[14]

Speed- Since C is a compiled language, the code is transformed into machine code before execution. This results in quicker execution than interpreted languages, which is critical for real-time voting systems.[15]

Portability: This code is easily portable between platforms with minimum change. This is necessary to ensure that the voting system can work on a wide range of operating systems and hardware combinations.[16]

Control over Memory: C gives direct memory management control, allowing developers to optimize memory utilization and eliminate memory leaks, maintaining the voting system's reliability.

Modification: C allows for extensive modification, allowing developers to modify the voting system to individual needs and limits.[17]

Security: Mini voting system enables for fine-grained control over system resources, which might be crucial when implementing security features like as access control, data encryption, and secure data storage in a voting system.[18]

# Weakness-

Difficulties with Technology: Electronic voting methods require dependable technological infrastructure, which can be difficult to build and maintain. The voting process could be delayed by technical faults or disturbances.

->Data Security: Maintaining data security and preventing leaks or the alteration of voter data or election results is a difficult task.

->Cost: Due to the need for software, hardware, security measures and maintenance needs, developing and implementing electronic voting systems can be expensive.

->User Training: In order to use electronic voting systems properly, both voters and election officials must be trained. It might be difficult to ensure that all stakeholders are comfortable with the technology.[19]

# Opportunities

->Faster Results: The time required to announce election results is reduced. This can lead to more timely and efficient elections.

->Easy Updates: Electronic systems are easily upgraded to fix the bugs or enhance functionality, ensure that the voting infrastructure is safe and up-to-date.

->Voter Education: Voters can be educated about candidates, problems and the method of voting by using electronic voting systems as teaching aids.

->Improved Accuracy: The inaccuracies that come with manual vote counting can be reduced by using electronic technology. In order to make sure that voters are cast correctly, they can additionally incorporate validation checks.

->Accessibility: Electronic voting can make the voting process more accessible for individuals with disabilities.[20]

# Threats

->Security Risks: Hacking and tampering are the two security risks that electronic voting systems are exposed to.

-> Resistance to Change: People who are concerned about security and the privacy and system reliability may oppose the switch to electronic voting.

->Regulatory Obstacles: For electronic voting systems, ensuring compliance with election rules and other regulatory standards can be difficult.

->Informational errors and falsehoods: False information can affect voters perceptions of and trust in the political process when it circulates online.

-> Lack of paper trail: Some electronic voting systems do not provide a paper trail, making it difficult to conduct post-election audits[21]

# Project Features

1. Modernizing the Voting System: Errors, frauds and inefficiency are some of the drawbacks of the old traditional paper-based voting system. This will eliminate these drawbacks and will be more accurate and convenient.
2. Enhanced Security: Security is enhanced by providing mechanisms like encryption, authentication mechanism for providing integrity and confidentiality of the voting process.
3. Efficiency & Speed: Tally of the votes and declaration of the results will be done in a much faster way, which will result in efficient and more timely elections.
4. Reduce Administrative Burden: Automating the task of counting votes and result tabulation will reduce the burden of the election officials which will lead to cost savings.[22]

Applications:

* Recording and Displaying Results: The system records votes and can calculate and display the election results, including the winning candidate and the total voter turnout.
* Educational Purposes: It can be used as a learning tool in computer science or programming courses to teach students about file handling, data structures, and user authentication in a practical context.
* Student Council Elections: The system can be used by educational institutions to conduct student council elections. Students can use their unique user IDs to cast their votes for their preferred candidates.[23]

# Design and Implementation Constraints

This code looks to be an electronic voting system implementation. It enables for an admin panel as well as a student panel to conduct and manage elections. The following are some important design and implementation considerations for this code:[24]

Global Variables: The code makes significant use of global variables to store information such as current election facts, candidate information, and student votes. While this facilitates data interchange, it may make the code more difficult to comprehend and maintain. The use of structures or classes to contain this data and functions to modify it may result in more modular and legible code.

File Management: The code uses file operations to store and retrieve election and vote data. File I/O can be sluggish and prone to errors. It is critical to appropriately handle file problems and to consider using a more resilient and efficient data storage option, such as a database.

Authentication of Users: For the admin panel, the code offers a simple user authentication system. While this is a simple approach, in a real-world scenario, a more secure and complex authentication system, such as username and password hashing, should be utilised.

Error Handling: More robust error handling should be included in the code, particularly when dealing with file operations. It currently lacks extensive error checking, which can result in unexpected behaviour if files are unable to be opened or read.

Security Issues: The code contains some security flaws. For example, the banID function automatically prohibits users based on their roll number. In practise, a more secure way of blocking users should be used. Furthermore, there appears to be no protection against multiple votes from the same person in the code.

Aspects of Modularity: Improved modularity would enhance the code. It would be easier to comprehend and maintain the code if it were divided into smaller, reusable functions with explicit responsibilities.

Code Remarks: There are no comments in the code that describe the purpose and operation of the various functions and blocks. Adding comments would make the code more understandable to others and easier to maintain in the future.

Validation of Data: The code does very little data validation. It is assumed that user IDs and vote inputs will always be in the expected format. To handle unexpected input, robust input validation and error management should be developed.

Integrity of Data:\* The code does not appear to take data integrity into account. Data integrity safeguards should be in place in a real-world election system to assure the correctness and trustworthiness of election outcomes.

\*Portability: \* The code appears to be written for a specific platform because it includes the non-portable conio.h> header. For user interaction, consider employing platform-independent libraries.

Code Effectiveness: The code may be more efficient, particularly when searching for and updating candidates' votes. It presently reads and writes full files for each vote, which can consume a lot of resources.

Handling Election Results: The code prints the election results to the console. Results should be saved and presented more securely and professionally in a real-world application.[25]

# Design diagram

# 

# Fig:1 (Output after running main.c)

# 

# Fig 2:(New election initiation)

# 

# 

# Fig 3: Creating candidates files

# 

# Fig:4 Login to student portal and casting votes

# 

# Fig 5: Casting votes

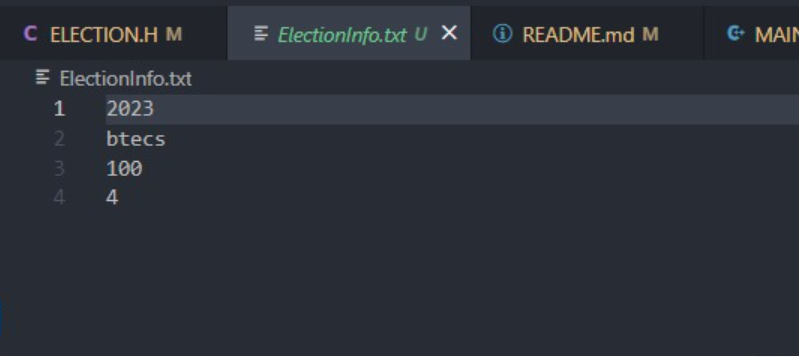


Fig 6: Election.txt file created

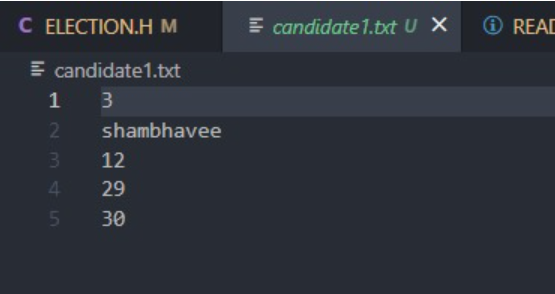


Fig 7: Candidate file

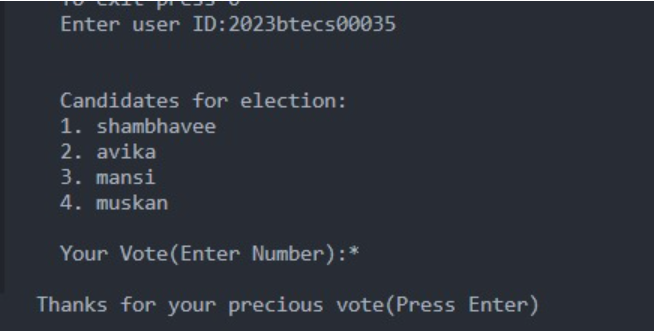


Fig 8: Names of all the candidates

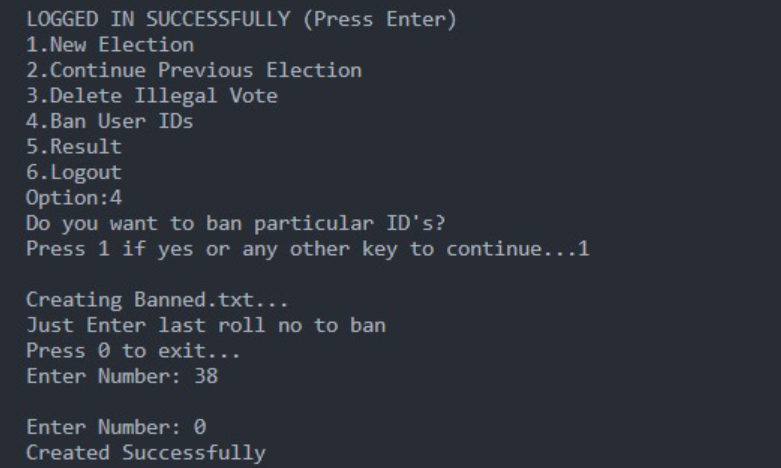


Fig 9: Creating Banned text file

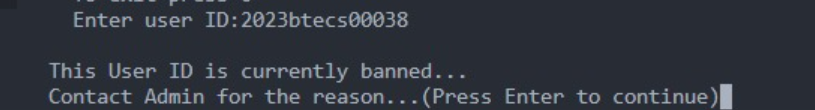


Fig 10: Banned user cannot cast the vote

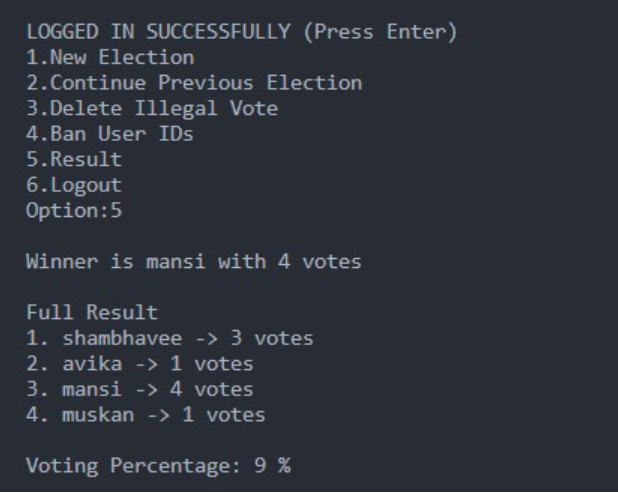
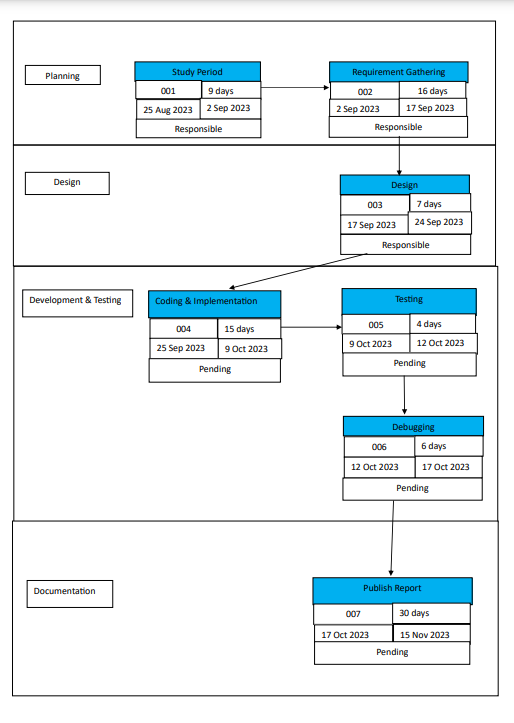


Fig 11: Final result display

**8. PERT Chart**

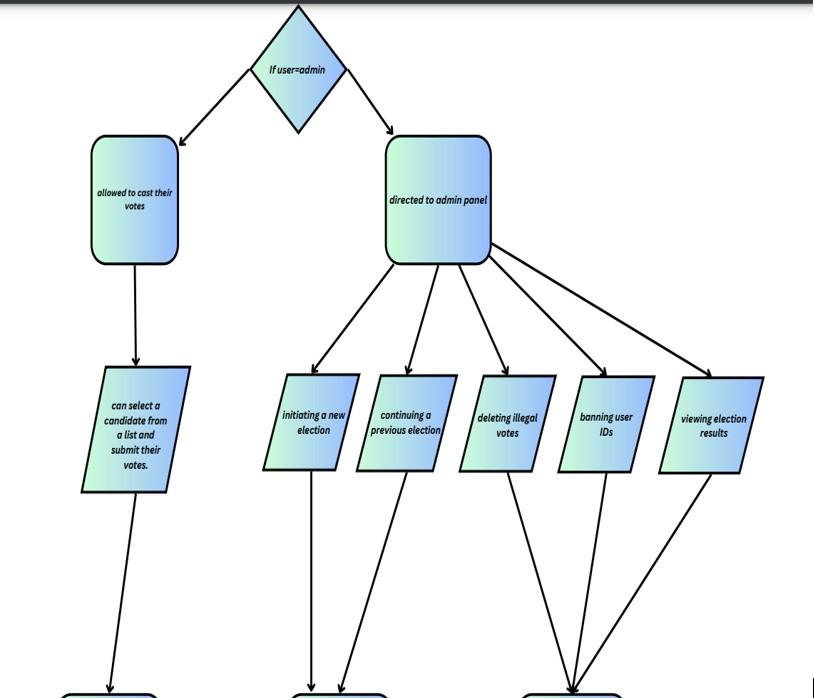


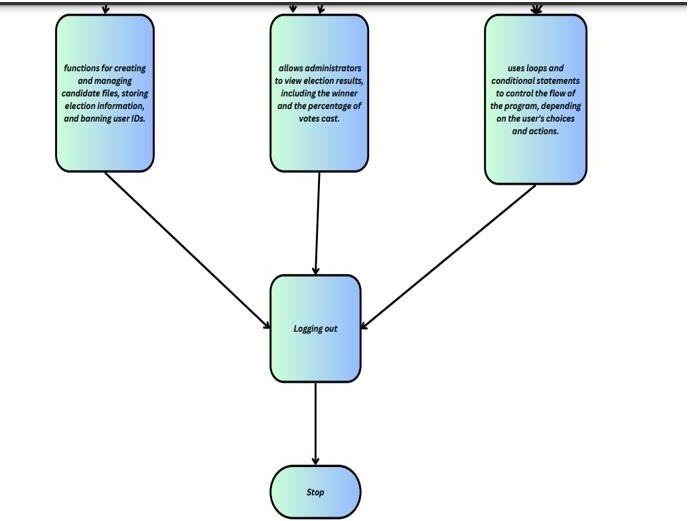
# 10.Programming Concept

This code appears to be a programme for running elections with two panels, one for administrators and one for students. Some of the programming principles and features used in this code are as follows:

1. Data Structures: The code stores information about elections, candidates, and voter IDs in a variety of data structures, including structs and arrays.
2. File Handling: The code reads and writes data to files in order to save information about candidates, elections, and banned user IDs.
3. User Authentication: For administrators, there is an authentication system that requires a username and password.
4. Input Validation: The code validates user IDs to ensure they match the anticipated format, year, and branch code. It also determines whether a person has previously voted or been banned.
5. Election Initialization: It allows the administrator to start a new election by entering information such as the election year, branch code, total number of voters, and number of candidates.
6. Vote Handling: It handles the voting process, which includes showing a list of candidates, accepting vote input from students, and saving the vote to the associated candidate's file.
7. Banning User IDs: Admins can ban individual user IDs, which are recorded in a "Banned.txt" file.
8. Election Results: The code can display election results, such as the winner and the amount of votes each candidate earned.
9. Error processing: The code has error checks and messages for a variety of circumstances, including file processing issues and incorrect inputs.
10. Looping and Menus: The code use loops to provide menus for both the admin and student panels, allowing users to conduct a variety of tasks.
11. Global Variables: The code employs global variables to hold data that must be accessed by various functions.
12. Function Calls: The code is organised into functions for various tasks, making it more modular and easier to maintain.
13. Structures and Typedef: The code specifies a structure for candidates and then uses typedef to generate a custom type (CANDIDATE) based on that structure.
14. String Manipulation: To handle strings and filenames, the code use string manipulation routines such as strcmp, strlen, and sprintf.
15. Conditional Statements: Conditional statements (if, else) are used for decision-making, such as validating administrator credentials and user IDs.
16. Input/Output: For user input and output, the code use printf and scanf.
17. Mathematical Operations: Mathematical operations are performed to obtain the year and roll number from user IDs.[26]

# Flow Chart





* 1. Technical Diagram

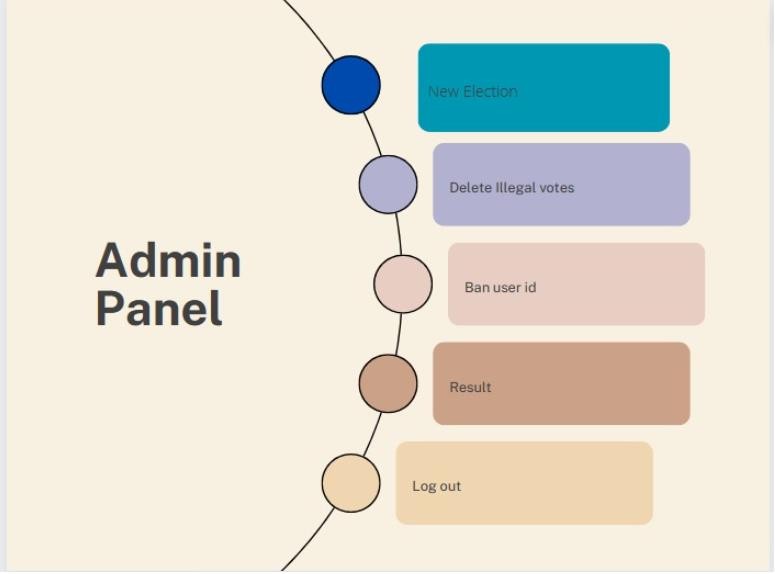


Fig 12: Feature of Admin Panel

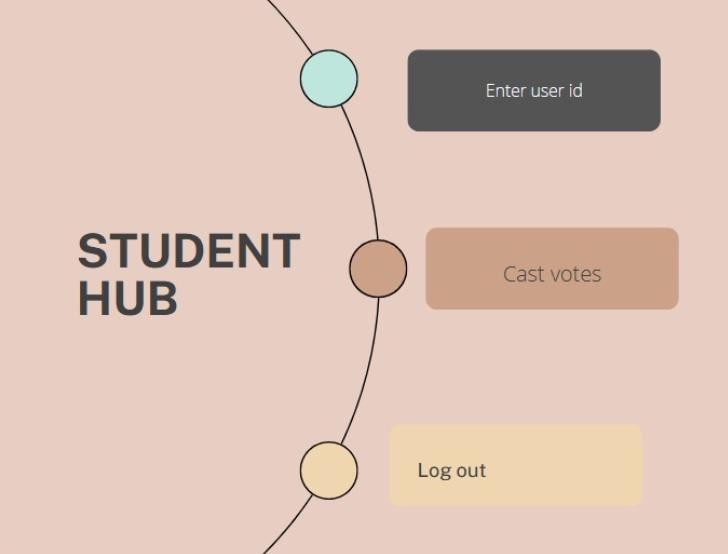


Fig 13: Feature of Student Hub

# System Requirements

The code you provided appears to be a simple electronic voting system. To run this code, you need a C compiler and a system that supports standard C libraries. Here are the system requirements for this code:

* + 1. \*\*C Compiler\*\*: You need a C compiler to compile and run the code. Common choices include GCC (GNU Compiler Collection) for Unix-like systems and Dev- C++ or Code::Blocks for Windows.
    2. \*\*Operating System\*\*: The code appears to be platform-independent and should run on most operating systems, including Windows, Linux, and macOS.
    3. \*\*Library Dependencies\*\*: This code relies on standard C libraries, including

`stdio.h`, `string.h`, and `stdlib.h`. Make sure your C compiler has these libraries available.

* + 1. \*\*Conio.h\*\*: The code includes `conio.h`, which is a non-standard library used for console I/O functions. It is commonly used in older Borland C compilers and DOS-based systems. Note that `conio.h` is not available in modern C compilers and is not part of the C standard library. You may need to make changes to your code to remove or replace `conio.h` with a different approach for handling input.
    2. \*\*File System\*\*: The code relies on file I/O to read and write election data. Ensure that the file system is accessible and that you have the necessary permissions to create, read, and write files.
    3. \*\*Hardware Requirements\*\*: The code itself does not have specific hardware requirements. It is a text-based program that should run on most modern and even older hardware configurations. However, the performance may vary depending on the number of candidates and voters, so consider the size of your input data.
    4. \*\*Console or Terminal\*\*: The code is designed to run in a console or terminal window. Make sure you have access to a command-line interface where you can interact with the program.
    5. \*\*User Input\*\*: Ensure you have a way to provide input to the program, such as a keyboard for entering user IDs and votes.
    6. \*\*Output Display\*\*: You should have a display to view program output and results. This can be a terminal window or console.
    7. \*\*Administrator and Student Access\*\*: You should have separate accounts for administrators and students to access the program as intended.
    8. \*\*Permissions\*\*: Ensure that you have the necessary permissions to create, read, and modify files on your system. The code creates and reads files like "ElectionInfo.txt" and "candidateX.txt," so permissions are essential.
    9. \*\*System Resources\*\*: The code does not appear to have high resource requirements. It should run on most systems with minimal CPU and memory usage.

Please note that the code uses certain functions and libraries that are not considered best practice in modern software development. It is advisable to replace non- standard libraries like `conio.h` with more standard and portable alternatives for better maintainability and compatibility. Additionally, consider improving the security of the system, as electronic voting systems must meet strict security requirements to ensure the integrity of the voting process.[27]

# Non-Functional Requirements

Non-functional requirements, often known as quality attributes, define the features of a software system that go beyond its basic functionality. They are critical for guaranteeing the overall performance, maintainability, and usability of the software. Here are several non- functional requirements and concerns in the code provided:

1. Performance: - The code should be efficient and responsive, especially while performing operations such as loading election information, saving votes, and banning user IDs.
2. Reliability: The code should handle faults and exceptions gracefully. It should not crash or generate unanticipated outcomes in the face of unexpected input or failures, such as failed file operations.
3. Security: Ensure that the code is safe from unauthorised access. User authentication is required for administrative tasks, but it must be secure against potential assaults.
   * Make certain that votes are securely recorded and that they cannot be tampered with.
   * Guard against SQL injection, file system flaws, and other typical security threats.
4. Scalability: The code should be scalable when the number of candidates, voters, or elections grows. It should be able to handle larger datasets and more candidates without significantly degrading performance
5. Maintenance: The code should be well-structured and well-documented to make future modifications and maintenance easier.
   * Proper commenting and documentation are essential for understanding.
6. Usability: The user interfaces (admin and student panels) should be simple and easy to use.
   * Error messages and user feedback should be clear and useful.
7. Portability: Ensure that the code is easily transferable or adaptable to other platforms and environments.
8. Data Integrity: Ensure that data is consistent and reliable by implementing data validation and data integrity checks.
9. Backup and Recovery: Implement data backup and recovery systems in the event of data loss or system failure.
10. Logging and Auditing: Implement critical event and action logging for auditing reasons. This can be useful for troubleshooting and tracking user behaviour.
11. Concurrency and Thread Safety: Ensure that the code can manage concurrent data and resource access without data corruption or race situations.[28]

# References

* 1. [https://www.studytonight.com/c-projects/election-system-project-using-](https://www.studytonight.com/c-projects/election-system-project-using-c-language) [c-language](https://www.studytonight.com/c-projects/election-system-project-using-c-language)
  2. <https://www.sciencedirect.com/science/article/pii/S0895717708001775>
  3. [https://www.lse.ac.uk/media@lse/research/EMTEL/Conference/papers/](https://www.lse.ac.uk/media%40lse/research/EMTEL/Conference/papers/Oostveen.pdf) [Oostveen.pdf](https://www.lse.ac.uk/media%40lse/research/EMTEL/Conference/papers/Oostveen.pdf)
  4. [https://www.slideshare.net/nitinbhasin3/online-voting-system-project-](https://www.slideshare.net/nitinbhasin3/online-voting-system-project-file) [file](https://www.slideshare.net/nitinbhasin3/online-voting-system-project-file)
  5. <https://www.geeksforgeeks.org/online-voting-system-in-c/>
  6. [https://www.academia.edu/7880540/\_MINI\_PROJECT\_REPORT\_ON\_ELEC](https://www.academia.edu/7880540/_MINI_PROJECT_REPORT_ON_ELECTRONIC_VOTING_MACHINE_BACHELOR_OF_TECHNOLOGY_IN_ELECTRICAL_AND_ELECTRONICS_ENGINEERING_Under_the_guidance_of) [TRONIC\_VOTING\_MACHINE\_BACHELOR\_OF\_TECHNOLOGY\_IN\_ELECTRIC](https://www.academia.edu/7880540/_MINI_PROJECT_REPORT_ON_ELECTRONIC_VOTING_MACHINE_BACHELOR_OF_TECHNOLOGY_IN_ELECTRICAL_AND_ELECTRONICS_ENGINEERING_Under_the_guidance_of) [AL\_AND\_ELECTRONICS\_ENGINEERING\_Under\_the\_guidance\_of](https://www.academia.edu/7880540/_MINI_PROJECT_REPORT_ON_ELECTRONIC_VOTING_MACHINE_BACHELOR_OF_TECHNOLOGY_IN_ELECTRICAL_AND_ELECTRONICS_ENGINEERING_Under_the_guidance_of)
  7. <https://www.irjet.net/archives/V4/i12/IRJET-V4I12256.pdf>
  8. [https://www.eballot.com/votes-and-elections/what-is-an-online-voting-](https://www.eballot.com/votes-and-elections/what-is-an-online-voting-system) [system](https://www.eballot.com/votes-and-elections/what-is-an-online-voting-system)
  9. <https://jpinfotech.org/online-election-system-online-voting-system/>
  10. <https://en.wikipedia.org/wiki/Voting_machine>
  11. [https://aceproject.org/ace-en/topics/va/201de-voting201d-and-external-](https://aceproject.org/ace-en/topics/va/201de-voting201d-and-external-voting) [voting](https://aceproject.org/ace-en/topics/va/201de-voting201d-and-external-voting)
  12. <https://www.shutterstock.com/search/electronic-voting>
  13. <https://www.yeselections.com/blog/top-features-online-voting-system>
  14. [https://www.slideshare.net/wilsonnandasaba/project-reportonline-](https://www.slideshare.net/wilsonnandasaba/project-reportonline-voting-system) [voting-system](https://www.slideshare.net/wilsonnandasaba/project-reportonline-voting-system)
  15. <https://dl.acm.org/doi/abs/10.1145/3439725>
  16. <https://orbilu.uni.lu/handle/10993/25419>
  17. <https://books.google.co.in/books?hl=en&lr=&id=AamdDQAAQBAJ&oi=fnd&pg=PA173&dq=mini+voting+system&ots=Sv79QTuT0O&sig=G1ZZbMIx_RQEjKDKwN8HJ-1nbx4&redir_esc=y#v=onepage&q&f=false>
  18. <https://books.google.co.in/books?hl=en&lr=&id=Dr96HqJUY2wC&oi=fnd&pg=PR11&dq=mini+voting+system&ots=3Z6AfjIB8D&sig=PeSTLs-S5wv1FJe8lqF4pYTP0PE&redir_esc=y#v=onepage&q=mini%20voting%20system&f=false>
  19. <https://jhalderm.com/pub/papers/diebold-ttbr07.pdf>
  20. <https://dspace.mit.edu/handle/1721.1/96589>
  21. <https://dl.acm.org/doi/abs/10.1145/3548606.3560563>
  22. <https://ieeexplore.ieee.org/abstract/document/5210119/>
  23. [https://www.slideshare.net/wilsonnandasaba/project-reportonline-](https://www.slideshare.net/wilsonnandasaba/project-reportonline-voting-system) [voting-system](https://www.slideshare.net/wilsonnandasaba/project-reportonline-voting-system)
  24. <https://dl.acm.org/doi/abs/10.1145/3548606.3560563>
  25. <https://www.shutterstock.com/search/electronic-voting>
  26. [https://www.eballot.com/votes-and-elections/what-is-an-online-voting-](https://www.eballot.com/votes-and-elections/what-is-an-online-voting-system) [system](https://www.eballot.com/votes-and-elections/what-is-an-online-voting-system)
  27. <https://books.google.co.in/books?hl=en&lr=&id=Dr96HqJUY2wC&oi=fnd&pg=PR11&dq=mini+voting+system&ots=3Z6AfjIB8D&sig=PeSTLs-S5wv1FJe8lqF4pYTP0PE&redir_esc=y#v=onepage&q=mini%20voting%20system&f=false>
  28. [https://www.slideshare.net/wilsonnandasaba/project-reportonline-](https://www.slideshare.net/wilsonnandasaba/project-reportonline-voting-system) [voting-system](https://www.slideshare.net/wilsonnandasaba/project-reportonline-voting-system)