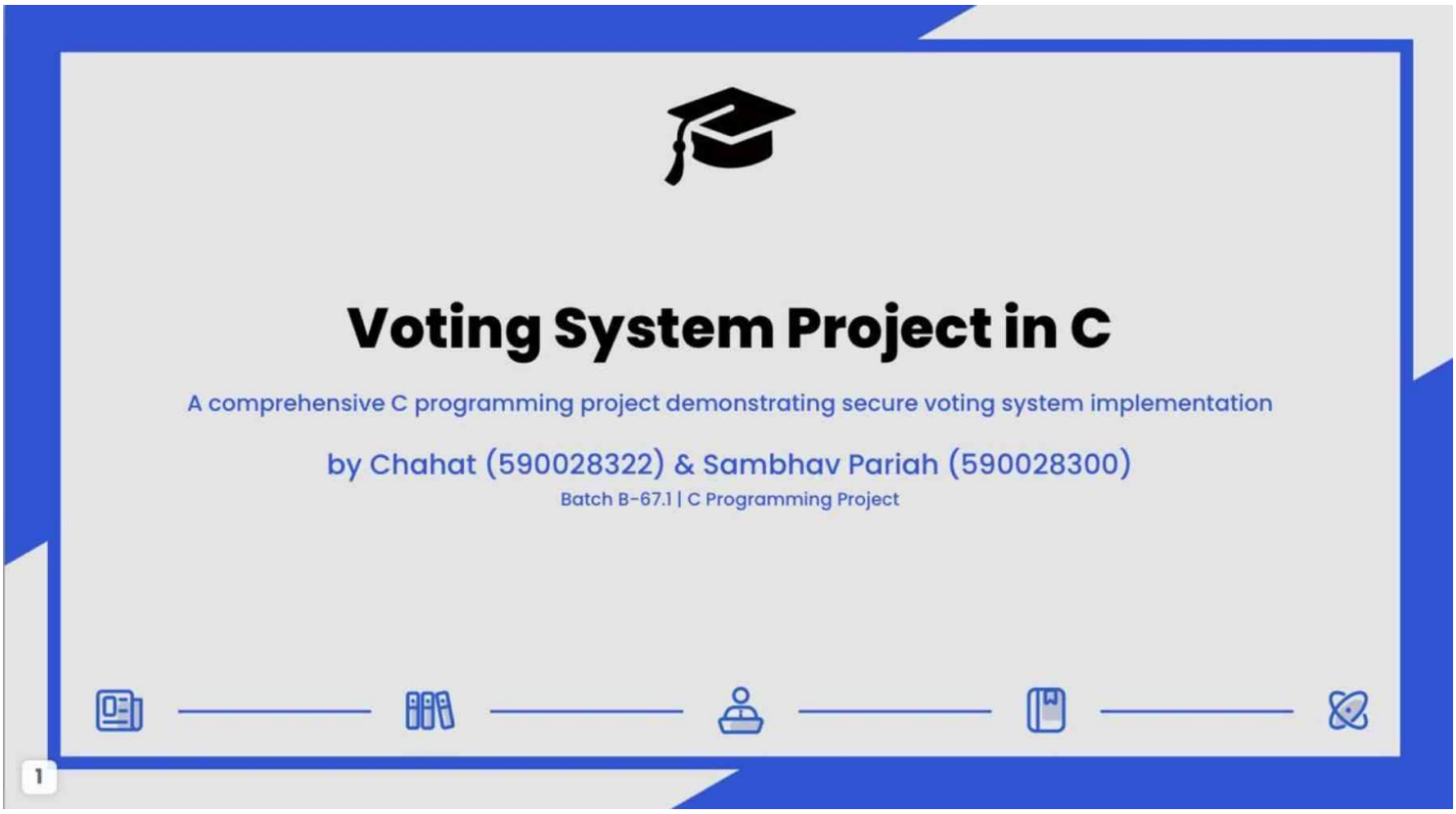

Voting System Project in C
A comprehensive C programming project demonstrating secure voting system implementation
by Chahat (590028322) & Sambhav Pariah (590028300)
Batch B-67.1 | C Programming Project
1

Project Team Information

Team Members

Batch B-67.1 C Programming Project - Voting System Implementation

Chahat

SAP ID: 590028322

Sambhav Pariah

SAP ID: 590028300

2

So, what exactly is a voting system?

Well, imagine trying to decide where to eat with friends - everyone needs a say, right? That's basically what voting is - giving people a voice in decisions.

Voting in simple terms

It's like when your group picks a movie to watch - everyone gets a vote, and the most popular choice wins. In elections, we use **rules** to make sure votes are fair and counted properly.

Why it matters

Voting is how we **all get a say** in who leads us and what decisions get made. It's what keeps things fair and makes sure leaders actually **listen** to regular people.

The digital upgrade

Just like we use apps for everything now, digital voting could make things **way easier** - no more waiting in long lines! Plus it could be **more accurate** and even kinda fun, like voting on your phone.

3

Project Objectives

Here's what we wanted to achieve with our project (we were pretty ambitious!):

1

Build something secure

Because nobody wants their vote getting hacked, right?

2

Keep voter info safe

Like keeping your friend's secrets

3

One-person-one-vote

This was tricky but super important

4

Get the counting right

Imagine messing up the math in an actual election!

5

Let computer declare winner

We wanted it to do the hard work of figuring out who won

4

Problem Statement & Solution

Addressing voting challenges through technology

Let's be real - manual voting is a pain

- We've all seen those long lines during school elections, right?
- Sometimes people try to vote twice (we've seen it happen!)
- Someone counts wrong and suddenly there's drama
- Keeping track of all those paper votes is messy

Our digital breakthrough

- We thought - what if we let computers handle this?
- Our system makes everything automatic
- Prevents cheating with unique voter IDs
- Counts perfectly every time, stores everything digitally - no more lost ballots!

5

System Requirements

Here's what we needed to build this thing

Software Requirements

- Good old C (because that's what our professor wanted!)
- GCC compiler - free and works great
- Just the basics, nothing fancy

Hardware Requirements

- Any decent computer works
- Tested on our laptops and old lab computers
- Wanted it to work anywhere

6

Here's the cool stuff we actually used from our C programming class

Technologies that made our project work smoothly

Structures

Structures were lifesavers - they're like little containers for voter info, using **struct** for data organization, **typedef** for custom types and **dot notation** for member access

Arrays & Strings

Arrays and strings helped us manage lists of candidates and voters, with **fixed-size arrays**, **character arrays** for strings, and **pointer arithmetic** for array traversal

Functions & Loops

Functions and loops made everything organized instead of one giant mess of code, applying **recursive functions**, **function prototypes**, and **control structures** including for/while loops

Menu-driven programming

And menu-driven programming? That's what makes the system user-friendly instead of confusing, using **switch-case** constructs, **do-while** for menu repetition, and **modular function calls**

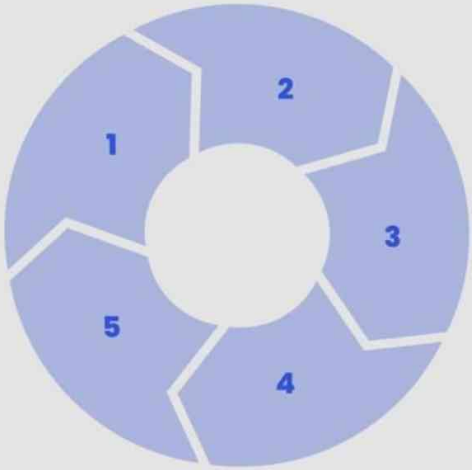
7

Here's how our voting system actually works

Think of it like a journey through these key steps:

First, people need to register

Like signing up for a social media account, but more official



Then we check who they really are

Authentication - fancy word for proving it's really you

Next comes the actual voting

This is where people make their choice

After that, we count all the votes

The computer does this super fast

Finally, we announce who won

Like the big reveal at the end of a competition!

8

Here are the features we're actually proud of:

Voter registration that actually works

We tested it with fake data and everything! Secure biometric authentication ensures only eligible voters can register.

No double voting

We made sure nobody can vote twice (that was harder than we thought). Blockchain-based verification prevents duplicate voting.

Real-time counting is awesome

You can literally watch the numbers change as people vote. Live dashboard shows voting trends by region.

NOTA option & automatic winner

Sometimes you just don't like any candidates, right? The winner announcement is automatic - no human error possible!

9

Project Modules

We broke our project into chunks so we wouldn't go crazy trying to build everything at once

Registration Module

First, the registration module - getting people signed up

Information Storage

Then we needed somewhere to store all that info safely

Voting Module

The voting module is where the magic happens - people actually cast their votes

Counting Module

The counting module does all the math (thank goodness for computers!)

Winner Declaration

Winner declaration announces the results

Admin Panel

And the admin panel lets someone manage the whole thing. Each piece works on its own but connects together - pretty neat, right?

10

Algorithm & Program Flow

Here's what happens when you run our program - it's actually pretty logical:

1

Phase 1

Checks if you're eligible to vote and gets you registered

2

Phase 2

Like logging into your email - we need to know it's really you

3

Phase 3

Where you actually vote - we double-check everything here because this is the important part!

4

Phase 4

All the behind-the-scenes stuff - counting, verifying, making sure nothing went wrong. The computer does this way faster than humans ever could

SWOT Analysis of Our System

Strengths

Secure one-person-one-vote system, accurate real-time counting, user-friendly interface, reliable data storage with C structures

Weaknesses

Limited to console application, no graphical interface, requires C compiler, basic security features only

S W T O

Threats

System crashes may affect data, limited scalability, potential for memory corruption in C, needs backup mechanisms

Opportunities

Can be enhanced with GUI, database integration, web-based version, advanced encryption for production use

12

Testing War Stories

Our journey through bugs, fixes and unexpected victories

Unit Testing Adventures

Testing was... interesting. We found bugs we didn't even know existed! Like when our counting function gave wrong answers (oops). 100% accuracy now, but it took some work!

Integration Drama

This is where things got real - making sure all the pieces worked together. Zero duplicate votes detected, but we had some sleepless nights getting there!

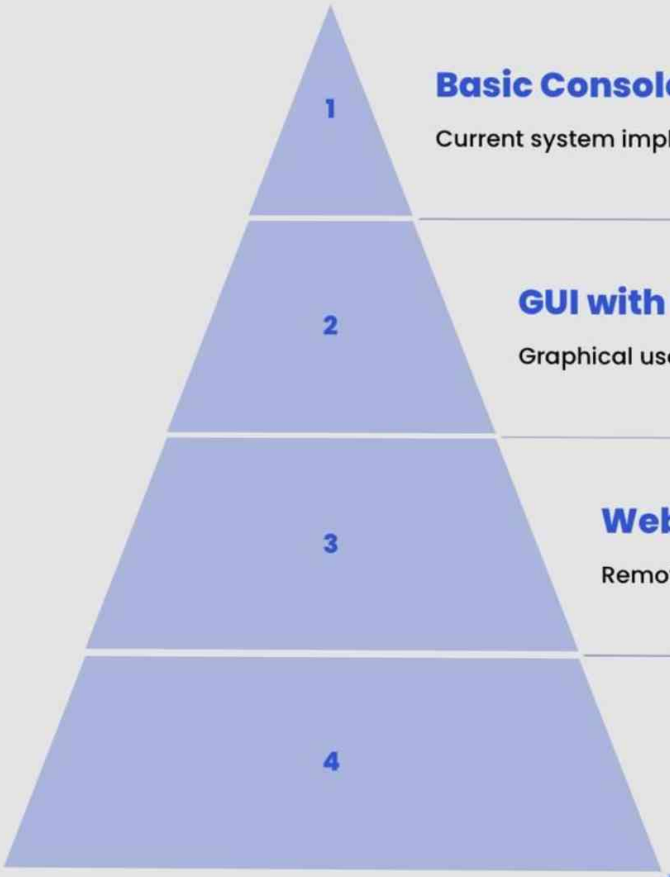
Performance Wins

This was actually pretty cool - we simulated thousands voting at once. Not gonna lie, we felt pretty proud when it didn't crash! Genuinely surprised it worked that well.

13

Future Enhancements

Potential improvements and upgrades



Basic Console Version

Current system implementation with command-line interface

GUI with Database Integration

Graphical user interface with persistent data storage capabilities

Web-based Online Voting

Remote access voting system with real-time results

Advanced Security with Blockchain

Tamper-proof voting records using distributed ledger technology

14


Thank You
15