# Requirement Gathering & Analysis for Voting Blockchain

Jenna Hall

# Overview

Project Name: Voting Blockchain Application

Type: Java Console Application

Goal: To securely record and validate votes using blockchain principles (immutability, transparency, security).

## Problem

Secure voting systems are critical. Traditional voting systems can be tampered with or lack transparency. A blockchain based voting system can provide a verifiable, tamper-resistant way to collect and validate votes (“Yay” or “Nay”).

## Target users

* Voter
  + Role: casts a “yay” or “nay” vote via the app
* Admin (Optional)
  + View vote tallies and blockchain integrity
* Single user for MVP (Console App)
* Future
  + Authenticated users

## Functional Requirements

FR1. The user can input a vote (“Yay” or “Nay”)

FR2. Each vote is added to a new block

FR3. Each block contains vote, timestamp, hash, and previous hash

FR4. The blockchain is stored in memory as a list

FR5. The system can verify if the blockchain is valid

FR6. The system prints out all votes in order

## Non-Functional Requirements

NFR1. Votes must be tampering proof using SHA-256 hashing

NFR2. Blockchain must reject incorrect hashes (invalid links)

NFR3. Application should handle invalid user input gracefully

## Scope of MVP

* In-Scope (V1)
  + Console-based vote submission
  + Vote stored as a block
  + Blockchain validation
  + Manual test cases
* Out-Of-Scope (Future)
  + GUI interface (JavaFX)
  + User authentication
  + Vote encryption or anonymity
  + Distributed ledger

## User Story

As a voter, I want to be able to cast a “Yay” or “Nay” vote so that it is permanently recorded in a secure blockchain for later use and can be verified at any time.

## Success Criteria

1. App starts and accepts vote

2. Votes are stored in a blockchain structure

3. Hashes validate the chain

4. Invalid input is handled

5. Blockchain data is readable and shows vote order