**Software Requirements**

**Homomorphic Voting System**

**Organization: Western Colorado University**

**Created By: Carter Auer, Joseph Vennard**

**1. Introduction**

**1.1 Purpose**

**(THE SOFTWARE WILL…)**

1.1.A: Be able to host and manage multiple polls created by users.

1.1.B: Support CRUD account operations, Including account creation, password changing, and standard login functionality.

1.1.C: Allow users to vote on polls.

1.1.D: Encrypt votes sent by users using homomorphic encryption.

1.1.E Display the encrypted vote values, along with the voters.

1.1.F: Allow any user, logged in or not, to view the current vote tally.

1.1.G Allow users to see previous votes and poll outcomes

1.1.H Allow users to view the outcomes of past votes, as well as the tally of current ones.

1.1.I: Allow users to edit their polls in the event of an error.

**1.2 Intended Audience**

1.2.A: The initial audience for the software is the Western Colorado University faculty and student body.

1.2.B: The audience following this could be scaled, depending on intention and performance.

**1.3 Intended Use**

**(THE SOFTWARE WILL…)**

**1.3.A: Provide a possible alternative to other voting systems as a secure, transparent, and adjustable method of polling groups.**

1.3.B: Allow anyone around campus to participate in decisions being made on campus and have a voice.

1.3.C: Give professors a tool to interact with students and get genuine input.

**1.4 Scope**

1.4.A: A web page accessible from browsers where users can create accounts, vote on polls, view current tallies and encrypted votes, and create their own polls.

1.4.B An implementation of Jeremy Muskat’s Homomorphic Voting Scheme for use by Western Colorado University students, faculty, and staff.

**1.5 Definitions and Acronyms**

Provide definitions the reader should understand before reading on.

DNS: Domain Name Service - something used as an easy to ready website name

HTTPS: Hypertext Transfer Protocol Secure - a secure website connection to keep account information safer

**2. Overall Description**

The system being built is an electronic voting web app with a security measure for users to have a truly private, secure vote based on an encryption system developed by Dr. Muskat.

**2.1 User Needs**

2.1.A: The user needs this software to be able to ensure their vote is kept private.

2.1.B: The user needs this software to show proof of voting.

**2.2 Assumptions and Dependencies**

2.2.A: We are assuming that each user will have an uninterrupted connection during their voting process.

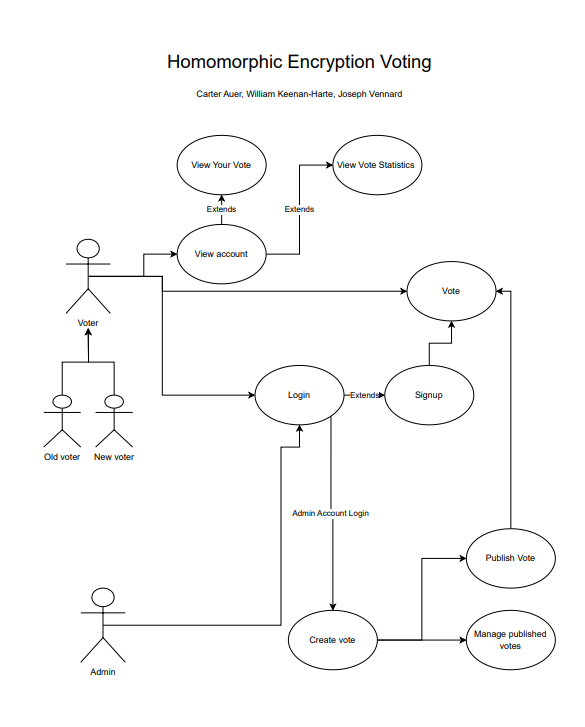
2.2.B: We are assuming that users will be confident in their final vote submission and will not try to change their answer after submitting their vote.

2.2.C: We are assuming we will have a database that is large enough to satisfy all necessary voters.

**3. System Features and Requirements**

**3.1 Functional Requirements**

3.1.A: Use cases:



3.1.B: Encryption of user vote

3.1.C: Decryption of total votes

**3.2 External Interface Requirements**

3.2.A: Flask database system

3.2.B: DNS hosting service

3.2.C: Python Anywhere

**3.3 System Features**

What features are required for the software to work?

3.3.A: Flask server

3.3.B: Flask database

**3.4 Nonfunctional Requirements**

3.4.A: HTTPS Security

3.4.B: Web app compatibility

3.4.C: Internet connection reliability

3.4.D: Scalable to voter quantity

Task Breakdown