  
  
 **Assignment Cover Sheet**

|  |  |
| --- | --- |
| **Subject Code:** CSIT321 |  |
| **Subject Name:** Graduation Project |  |
| **Submission Type:** Turnitin |  |
| **Assignment Title:** Final Document |  |
| **Student Name:** Ayesha Riyaz, Saher Hasan, Farhana Islam, Mahmoud AlSofyani |  |
| **Student Number:** 5578280, 5658500, 5628234, 5605635 |  |
| **Student Phone/Mobile No.** 0567613937 |  |
| **Student E-mail:** [fi982@uowmail.edu.au](about:blank), [sh392@uowmail.edu.au](about:blank), [sasmm065@uowmail.edu.au](about:blank) , [ar727@uowmail.edu.au](about:blank) |  |
| **Lecturer Name:** Dr. May El Barachi , Dr. Zeenath Reza Khan |  |
| **Due Date:** 14/05/2020 |  |
| **Date Submitted:** 14/05/2020 |  |

|  |  |
| --- | --- |
| **PLAGIARISM:** The penalty for deliberate plagiarism is FAILURE in the subject. Plagiarism is cheating by using the written ideas or submitted work of someone else. UOWD has a strong policy against plagiarism.  The University of Wollongong in Dubai also endorses a policy of non-discriminatory language practice and presentation.  **PLEASE NOTE:**STUDENTS MUST RETAIN A COPY OF ANY WORK SUBMITTED | **DECLARATION:** I/We certify that this is entirely my/our own work, except where I/we have given fully-documented references to the work of others, and that the material contained in this document has not previously been submitted for assessment in any formal course of study. I/we understand the definition and consequences of plagiarism.  **Signature of Student:** |

|  |  |  |
| --- | --- | --- |
| |  | | --- | | **Optional Marks:** | | **Comments:** | |

https://my.uowdubai.ac.ae/images/scissors.gif

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | **Lecturer Assignment Receipt**(To be filled in by student and retained by Lecturer upon return of assignment) | | | **Subject:** | **Assignment Title:** | | **Student Name:** | **Student Number:** | | **Due Date:** | **Date Submitted:** | | **Signature of Student:** | | |

https://my.uowdubai.ac.ae/images/scissors.gif

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | **Student Assignment Receipt** (To be filled in and retained by Student upon submission of assignment) | | | **Subject:** | **Assignment Title:** | | **Student Name:** | **Student Number:** | | **Due Date:** | **Date Submitted:** | | **Signature of Lecturer** | | |



**TasweetX – Blockchain Voting System**

This technical document has been prepared as part of requirements for the capstone graduation project subject CSIT321

*From University of Wollongong in Dubai*

*Prepared by*:

**Team F.A.M.S**

Ayesha Riyaz (5578280)

Saher Hasan (5658500)

Farhana Islam (5628234)

Mahmoud AlSofyani (5605635)

*Spring 2020 Batch*

# CERTIFICATION

We, Team FAMS, declare that this technical document, submitted in fulfilment of the requirements for the completion of the graduation project subject CSIT321, in the Faculty of Engineering & Information Sciences, University of Wollongong in Dubai, is wholly our own work unless otherwise referenced or acknowledged. The document has not been submitted for qualifications at any other academic institution.

Ayesha Riyaz

Saher Hasan

Farhana Islam

Mahmoud AlSofyani

# EXECUTIVE SUMMARY

Every organization around the world that relies on a voting system is susceptible to tampering. This means foul play can be involved in presidential elections, cabinet approvals, and so on. TasweetX aims to utilize the blockchain domain to ensure all votes are timestamped, verifiable, encrypted, and stored permanently. Because blockchain uses encryption to link each block, all votes are linked to each other. If a vote was to be modified, that is recorded as an entirely new block. This allows voters to ensure their votes are protected from any unauthorized tampering.

# ACKNOWLEDGEMENTS

We have taken immense efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.

We are highly indebted to University of Wollongong and the faculty of Engineering and Information Sciences and our course coordinator Dr Zeenath Reza Khan, for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support.

We would like to express our gratitude towards our parents & mentor of this project, Dr May ElBarachi (faculty of EIS in UOWD) for her kind co-operation and encouragement which helped us in completion of this project.

Our thanks and appreciations also go to our colleagues in developing the project and people who have willingly helped us out with their abilities, and constructive criticisms

Contents

[CERTIFICATION 1](#_Toc40394852)

[EXECUTIVE SUMMARY 2](#_Toc40394853)

[ACKNOWLEDGEMENTS 3](#_Toc40394854)

[I. Introduction 9](#_Toc40394855)

[Innovation in your project field eg. education/automobile) 9](#_Toc40394856)

[Problem Area – problem statement 9](#_Toc40394857)

[Background study 9](#_Toc40394858)

[Our solution/approach 10](#_Toc40394859)

[Summary of the report 11](#_Toc40394860)

[II. Feasibility 11](#_Toc40394861)

[Project Charter 11](#_Toc40394862)

[Scope Statement 12](#_Toc40394863)

[PESTEL 13](#_Toc40394864)

[Work Breakdown Structure 13](#_Toc40394865)

[Project Budget 14](#_Toc40394866)

[Performance Measurement Baselines 14](#_Toc40394867)

[Milestones and Associated Dates 15](#_Toc40394868)

[Staffing 16](#_Toc40394869)

[Risk Management Plan 17](#_Toc40394870)

[Qualitative Risk Analysis: 17](#_Toc40394871)

[Probability/Impact Matrix: 18](#_Toc40394872)

[Open Issues 18](#_Toc40394873)

[Division of work among team members 18](#_Toc40394874)

[III. Requirements 19](#_Toc40394875)

[Introduction 19](#_Toc40394876)

[Overview 19](#_Toc40394877)

[Goals and Objectives 19](#_Toc40394878)

[Scope 20](#_Toc40394879)

[Definitions 20](#_Toc40394880)

[Document Conventions 20](#_Toc40394881)

[Assumptions 21](#_Toc40394882)

[General Design Constraints 21](#_Toc40394883)

[Product Environment 21](#_Toc40394884)

[User Characteristics 21](#_Toc40394885)

[Mandated Constraints 21](#_Toc40394886)

[Potential System Evolution 22](#_Toc40394887)

[Non-functional Requirements 22](#_Toc40394888)

[Usability Requirements 22](#_Toc40394889)

[Operational Requirements 23](#_Toc40394890)

[Hardware Requirements 23](#_Toc40394891)

[Software Requirements 23](#_Toc40394892)

[Performance Requirements 23](#_Toc40394893)

[Security Requirements 24](#_Toc40394894)

[Voters: 24](#_Toc40394895)

[Candidates: 24](#_Toc40394896)

[Safety Requirements 25](#_Toc40394897)

[Legal Requirements 25](#_Toc40394898)

[Other Quality Attributes 25](#_Toc40394899)

[Documentation and Training 25](#_Toc40394900)

[External Interface 26](#_Toc40394901)

[User Interface 26](#_Toc40394902)

[System Features 26](#_Toc40394903)

[Functional Requirements 26](#_Toc40394904)

[Voters & Candidates 26](#_Toc40394905)

[Regulatory Authorities: 30](#_Toc40394906)

[IV. Design 36](#_Toc40394907)

[Introduction 36](#_Toc40394908)

[System Overview 36](#_Toc40394909)

[Web Application 36](#_Toc40394910)

[Core Blockchain 36](#_Toc40394911)

[Design Map 36](#_Toc40394912)

[Definitions and Acronyms 36](#_Toc40394913)

[Design Assumptions 37](#_Toc40394914)

[Assumptions 37](#_Toc40394915)

[Constraints 37](#_Toc40394916)

[System Environment 38](#_Toc40394917)

[Design Methodology 38](#_Toc40394918)

[Risks and Volatile Areas 38](#_Toc40394919)

[Architecture 39](#_Toc40394920)

[Overview 39](#_Toc40394921)

[Rationale 39](#_Toc40394922)

[High-Level Design 39](#_Toc40394923)

[Conceptual View 39](#_Toc40394924)

[Wireframes 40](#_Toc40394925)

[Voters 40](#_Toc40394926)

[Sign Up, sign in and view/edit profile 40](#_Toc40394927)

[View Candidate Dashboard and Cast Vote 41](#_Toc40394928)

[Regulatory Authorities 41](#_Toc40394929)

[Start/end election 41](#_Toc40394930)

[Add candidate 42](#_Toc40394931)

[Delete candidate 42](#_Toc40394932)

[Announce results 42](#_Toc40394933)

[Update Candidate Profile 42](#_Toc40394934)

[Low-Level Design 43](#_Toc40394935)

[Data Dictionary 43](#_Toc40394936)

[Off-Chain 43](#_Toc40394937)

[On-Chain 44](#_Toc40394938)

[Use-Case Diagrams 45](#_Toc40394939)

[Voters/Candidates 45](#_Toc40394940)

[Regulatory Authorities 46](#_Toc40394941)

[Class Diagrams 46](#_Toc40394942)

[Voters 48](#_Toc40394943)

[Regulatory Authorities 49](#_Toc40394944)

[CRC Cards 50](#_Toc40394945)

[Sequence Diagrams 52](#_Toc40394946)

[Voter 52](#_Toc40394947)

[Regulatory Authority 52](#_Toc40394948)

[Activity Diagrams 53](#_Toc40394949)

[Voters 53](#_Toc40394950)

[Regulatory Authorities 54](#_Toc40394951)

[User Interface Design 55](#_Toc40394952)

[Website Control (Voter’s Side) 55](#_Toc40394953)

[Sign In & Create Account 55](#_Toc40394954)

[Candidate Dashboard and Vote Page 56](#_Toc40394955)

[Voters Profile and Edit Profile 57](#_Toc40394956)

[Web Portal (Regulatory Authority’s Side) 57](#_Toc40394957)

[Log In 58](#_Toc40394958)

[Start Elections 58](#_Toc40394959)

[End Elections 59](#_Toc40394960)

[Add Candidates 60](#_Toc40394961)

[Delete Candidate 62](#_Toc40394962)

[Edit Candidate Profile 63](#_Toc40394963)

[Announce Results 64](#_Toc40394964)

[Help Page 65](#_Toc40394965)

[V. Implementation 66](#_Toc40394966)

[Pre-requisites 66](#_Toc40394967)

[Development Environment 66](#_Toc40394968)

[Installing Hyperledger Fabric 66](#_Toc40394969)

[Developing the Blockchain Solution 67](#_Toc40394970)

[VI. Testing 68](#_Toc40394971)

[Functional Testing Goals 68](#_Toc40394972)

[Functional Test Plan Scope 68](#_Toc40394973)

[In Scope 68](#_Toc40394974)

[Out of Scope 68](#_Toc40394975)

[Functional Test Plan Assumptions and Constraints 68](#_Toc40394976)

[Assumptions 68](#_Toc40394977)

[Constraints 68](#_Toc40394978)

[Functional Test Team Roles and Responsibilities 68](#_Toc40394979)

[Functional Requirement Test Cases 69](#_Toc40394980)

[1.1 User (Voter) requirements 69](#_Toc40394981)

[1.1.1 Login 69](#_Toc40394982)

[1.1.2 Logout 71](#_Toc40394983)

[1.1.3 Create Account 72](#_Toc40394984)

[1.1.4 Manage Profile Page 73](#_Toc40394985)

[1.1.5 Candidate Dashboard Page 74](#_Toc40394986)

[1.1.6 Selected Candidate Page 74](#_Toc40394987)

[1.1.7 Voter Dashboard Page 76](#_Toc40394988)

[1.2 Regulatory Authority requirements 77](#_Toc40394989)

[1.2.1 Login 77](#_Toc40394990)

[1.2.3 Start Election 79](#_Toc40394991)

[1.2.4 End Elections 80](#_Toc40394992)

[1.2.5 Add Candidate 80](#_Toc40394993)

[1.2.6 Delete Candidate 82](#_Toc40394994)

[1.2.7 Edit Candidate Profile 82](#_Toc40394995)

[1.2.8 Tally Votes 83](#_Toc40394996)

[1.2.9 Announce Results 84](#_Toc40394997)

[Functional Testing Results 85](#_Toc40394998)

[Traceability Matrix 86](#_Toc40394999)

[Traceability Matrix for Voters’ Requirements 86](#_Toc40395000)

[Traceability Matrix for RA’s Requirements 86](#_Toc40395001)

[Maintenance 87](#_Toc40395002)

[Appendices 88](#_Toc40395003)

[Feasibility: 88](#_Toc40395004)

[Gantt Chart 88](#_Toc40395005)

[Requirements: Revision History 89](#_Toc40395006)

[Requirements Traceability Matrix 89](#_Toc40395007)

[Qualitative Risk Analysis 90](#_Toc40395008)

[Design: 91](#_Toc40395009)

[Revision History 91](#_Toc40395010)

[Testing: 92](#_Toc40395011)

[Requirement list 92](#_Toc40395012)

[Testcase List 92](#_Toc40395013)

[Skill Tracker 93](#_Toc40395014)

[Meeting Minutes 95](#_Toc40395015)

[References 111](#_Toc40395016)

# Introduction

Computer Systems, Importance in 21st century – fourth industrial revolution

As the 21st century emerges forward, it is undeniable that technology has made astounding developments ever since it was discovered. There was a time when computer systems were only heard or thought of as a mere dream and luxury. However, today most of us if not all, cannot imagine our lives without them since it has become the basic necessity of our lives. We now live in a world where The Fourth Industrial Revolution has embedded into our lives by rapidly altering the way we live to how we work and communicate. In fact, it is seen to have reshape the systems such as education, government, healthcare, and commerce—almost every aspect of our life.

The fourth industrial revolution also involves the usage of new emerging technologies such as artificial intelligence, blockchain, the Internet of Things and smart materials among many others, which are redefining the boundary between the digital and physical worlds. With no doubt, that these powerful new technologies are being developed and implemented at an expeditious rate and will also continue to do so in the future as it continues to affect human identities, societies and political systems, we can say that it has given us a vision like opportunity and even more responsibility to give it a meaningful structure and purpose in ways that can positively impact the world.

## Innovation in your project field eg. education/automobile)

Our project, TasweetX, is a voting system that changes the traditional method of voting by making it more reliable and safer for both the voters and candidates. With the breakthrough innovation, that is the blockchain technology (domain) is utilized to ensure all votes are verifiable and stored permanently, making voting for any election seamless and secure. Hence, the innovation lies in the political electoral system of our project field.

## Problem Area – problem statement

Since the advent of democracy, elections have seen a multitude of voting mechanisms, ranging from paper ballots to EVMs. Yet, none of the mechanisms have been free from manipulations or tampering. History has also shown the flaws of current ways of election that have led to EVM tampering, voter ID fraud, inaccurate electoral rolls and voter impersonation to name a few. Thus, It is high time for us to start coming up with better and smarter ways to try and change this traditional method of voting. With the objective to

change the traditional method of voting, blockchain provides a safe and comparatively secure method to eliminate these shortcomings and to conduct elections with the hope to make elections unhackable.

## Background study

In the year 2016, people in Arizona had to wait as long as five hours just to vote in a primary election. Besides the extensive amount of time wasted standing in ques, other problems also remain unresolved. Such as:

● Technical: possibility of ballot stuffing; use of fading ink to fake results; fake protocols; violation of secrecy; intentional errors in calculations; the slow speed of vote counting.

● Social: bribery, threatening for life or administrative pressure on voters.

● Economic: the high cost to the state budget. For example, the 2015 elections in Russia cost almost $100 million. Electronic Voting Machines also have their own vulnerabilities like:

● Security issues (electronic systems are vulnerable to hacking, usually followed along with unpleasant scandals)

● Certifying election results (as opposed to paper ballots that can be counted, accuracy and trust issues)

● Unintentional Error (probability of incorrect system operation due to software errors)

Our proposed solution implements Blockchain to overcome these issues. Blockchain makes the election process digital, anonymous and decentralized instead. Besides, the use of the blockchain provides additional benefits:

1. Validity: Voting results organized using blockchain technology cannot be faked. An authorized voter can always check how many votes were issued after casting her or his vote, the process of distribution to wallets, and the time the transactions were conducted.

2. Transparency: Blockchain provides the opportunity to control the voting process, since any interested person can deploy a node with a full copy of all data and independently analyse them at the blockchain level.

3. Anonymity: Every voter will have the opportunity to create a pair of public and private keys on a local machine and none except that voter will know that a particular wallet belongs to her or him.

4. Speed of data processing: Voting, counting the votes and then tallying makes the whole election a cumbersome process. Voting is conducted within a city, district, region, country and the costs associated with it is significant. Moreover, there are organizational difficulties, temporary loss curtails, where voting and data. Decentralization of blockchain will let the results of voting to be shared throughout the country, holistically made visible, even though each region or district or city can operate its own system node for load distribution. (Sergeenkov, 2019)

## Our solution/approach

Our approach to the problem (project) is making voting simple for voters and trustworthy and as legitimate as possible for both voters and candidates. Voting for a candidate can be as simple as visiting a website. Voters would have to login and verify their identity, before they are issued their digital id, making voting completely anonymous. This also ensures that the voter is an actual human being. The voter database will be distributed among everyone. No single authority will control the elections. With the election so transparent, each voter can open the ballot box database, locate and check their vote remains unaltered. By tallying all these information even illegitimate votes can be prevented.

Our blockchain voting system, TasweetX, will allow voters to cast votes to their chosen candidates based on the Emirate they are in. Regulatory Authorities will be responsible for managing candidates and control the activation and deactivation of the election period.

## Summary of the report

– what the report/documentation contains and in what order

# Feasibility

## Project Charter

|  |
| --- |
| Project Name: TasweetX |
| Project Start Date: September 15, 2019  Project Finish Date: May 3, 2019 |
| Key Schedule Milestones:  · Code completion Feb 15  · Prototype of the system March 1  · Design of the complete system April 1  · Beta Testing April 15 |
| Budget Information:  AED 2 Million |
| Project Members:  Team Leader : Ayesha Riyaz  Scribe : Saher Hasan  Mahmoud AlSofyani  Farhana Islam |
| Project Objectives:  Every year, elections are conducted in some parts of the world. While democracy was established to establish trust and giving power and voice to the normal citizens, the voting mechanism still has some loopholes to be fixed.  TasweetX is developed to solve all the voting issues and establish trust in the process. |
| Main Project Success Criteria: The software requirements must meet all the written specifications, tested carefully and completed within the required timespan of eight months. Computational power is also kept in minimum. |
| Approach:  · Authenticate the users and keep updating the user database  · Issue a request to quote for hardware and software  · Use internal staff for analysis and mentoring |

## Scope Statement

Since the advent of democracy, elections have seen a multitude of voting mechanisms, ranging from paper ballots to EVMs. Yet, none of the mechanisms have been free from manipulations or tampering. History has also shown the flaws of current ways of election that have led to EVM tampering, voter ID fraud, inaccurate electoral rolls and voter impersonation to name a few. With the objective to change the traditional method of voting, blockchain provides a safe and comparatively secure method to eliminate these shortcomings and to conduct elections with the hope to make elections unhackable.

## PESTEL

Political Factors:   
 -Stakeholders (voters, Elected Candidates for Voting) and their demands  
 -Manipulation of votes will be impossible

Economic Factors:  
 -Unemployment Level may increase due to removal of middlemen

Social Factors:   
 -Ethical Factors like causing people to lose their jobs  
 -Media involvement  
 -The regular attitude of people involved may change  
 -General Trends involved with Voting in the past may change

Technological Factors:  
 -More advanced technology needed  
 -Higher investments required  
 -May require licensing if the system is modified for use of election voting, etc.

Environmental Factors:  
 -Remove use of papers used in ballot voting- more environment friendly

Legal Factors:  
 -University rules and regulations need to be considered while testing and implementing the system

## Work Breakdown Structure

All the components for this system will be software based. Therefore, many of the components will require timely execution and completion for the entire system to be delivered on time.

TasweetX requires a backend management system, a front end, and the blockchain system itself.

Phase 1 - Research & Training (1 Month & 22 Days)

* Research on Blockchain Development and Tools (31 days)
* Research on Smart Contracts (7 Days)
* Learn programming languages needed (14 Days)

Phase 2 - Design (1 Month & 28 Days)

* Generate Requirements (7 Days)
* System Constraints (7 Days)
* Determining Risks (5 Days)
* System Architecture (21 Days)
* UML Diagrams (14 Days)
* Design UI - Website (4 Days)

Phase 3 - Implementation (1 Month & 19 Days)

* Coding the first block for this system using Hyper ledger fabric (14 Days)
* Coding the smart contracts (14 Days)
* Implementation of UI - Website (Regulatory authority pages)(7 Days)
* Implementation of UI - Website (Voter pages) (14 Days)

Phase 4 - Testing (23 Days)

* Testing core system (7 Days)
* Testing UI systems - Website (Regulatory authority pages)(4 Days)
* Testing UI system - Website (Voter pages) (4 Days)
* Test prototype (8 Days)

*Total estimated time for completion of project: 6 Months*

## Project Budget

|  |  |  |
| --- | --- | --- |
| ***Project Element*** | ***Constraint*** | ***Estimate*** |
| Programming Languages Training | Time | 1 Month |
| Testing and re-testing | Time | 2 Months |
| Computers | Cost | AED 8K / computer |

## Performance Measurement Baselines

Based on preliminary discussions and planning, we expect to complete the system including research, training, testing and retesting in a total of 5 months. Gantt chart is provided as Appendix 1.

Four phases are planned for this system.

* Research & Training
* Design
* Implementation
* Testing

## Milestones and Associated Dates

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr | Task | Duration/days | Start Date | End Date |
| 1 | Research on Blockchain Development and Tools | 31 | 22-Sep-19 | 23-Oct-19 |
| 2 | Research on Smart Contracts | 7 | 23-Oct-19 | 30-Oct-19 |
| 3 | Learn programming languages needed | 14 | 30-Oct-19 | 13-Nov-19 |
| 4 | Generate Requirements | 7 | 13-Nov-19 | 20-Nov-19 |
| 5 | System Constraints | 7 | 20-Nov-19 | 27-Nov-19 |
| 6 | Determining Risks | 5 | 27-Nov-19 | 02-Dec-19 |
| 7 | System Architecture | 21 | 02-Dec-19 | 23-Dec-19 |
| 8 | UML Diagrams | 14 | 23-Dec-19 | 06-Jan-20 |
| 9 | Design UI - Website | 4 | 06-Jan-20 | 10-Jan-20 |
| 10 | Coding the first block for this system using Hypder ledger fabric | 14 | 10-Jan-20 | 24-Jan-20 |
| 11 | Coding the smart contracts | 14 | 24-Jan-20 | 07-Feb-20 |
| 12 | Implementation of UI - Website (Regulatory Authority) | 7 | 07-Feb-20 | 14-Feb-20 |
| 13 | Implementation of UI - Website (Voters) | 14 | 14-Feb-20 | 28-Feb-20 |
| 14 | Testing core system | 7 | 28-Feb-20 | 06-Mar-20 |
| 15 | Testing UI systems - Website (Regulatory Authority) | 4 | 06-Mar-20 | 10-Mar-20 |
| 16 | Testing UI systems - Website (Voters) | 4 | 10-Mar-20 | 14-Mar-20 |
| 17 | Test Prototype | 8 | 14-Mar-20 | 22-Mar-20 |

## Staffing

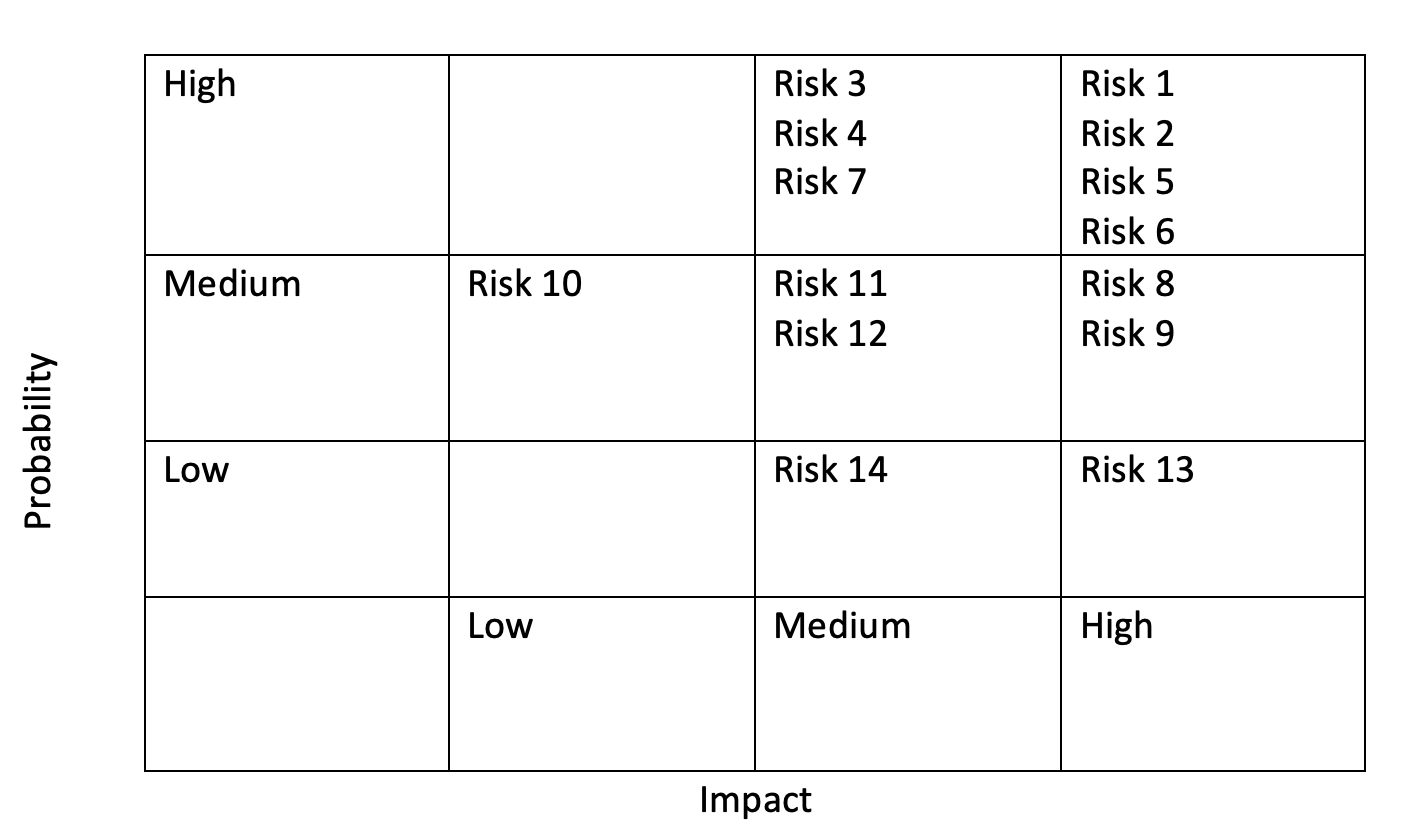
|  |  |  |  |
| --- | --- | --- | --- |
| Member/Person | Role | Cost | Available Dates |
| Ayesha Riyaz | Research, Documentation, Developer (Blockchain, UI Website) | none | Full Time- September 2019 to May 2020 |
| Saher Hasan | Research, Documentation, Developer (Blockchain, UI Website) | none | Full Time- September 2019 to May 2020 |
| Farhana Islam | Research, Documentation, Developer (Blockchain, UI Website) | none | Full Time- September 2019 to May 2020 |
| Mahmoud AlSofyani | Research, Documentation, Developer (Blockchain, UI Website) | none | Full Time- September 2019 to May 2020 |
| Dr May ElBarachi | Mentor Team during development and implementation of project | none | University Timings- during consultation hours Scheduled appointments and meetings weekly |
| Dr Halim Khelalfa | References and additional advice for Blockchain | none | University Timings- during consultation hours By appointment, when needed |

## Risk Management Plan

## Qualitative Risk Analysis:

The qualitative risk analysis is listed in the appendix.

## Probability/Impact Matrix:



## Open Issues

One of the issues that requires further attention is the security vulnerabilities associated when building an online and real time system. However, to find ways to overcome or eradicate any risk and vulnerabilities of the system that we are building, we intend to research a lot more in depth in maximum of one weeks’ time.

## Division of work among team members

|  |  |
| --- | --- |
| Team Member | Contribution |
| Ayesha Riyaz | Project Scope, Project Charter, Requirements Matrix |
| Mahmoud AlSofyani | Work breakdown structure,  Project Budget Analysis,  Performance measurement baselines |
| Farhana Islam | Risk Management Plan,  Open Issues |
| Saher Hasan | Milestones and Associated dates,  Staffing, PESTEL Analysis |

# Requirements

## Introduction

## Overview

This document defines the requirements for the TasweetX voting application. TasweetX is being developed to ensure voting elections are fraud-free and that all votes are accounted for. The purpose of this document is to represent TasweetX system requirements in an appropriate level of understanding so clients and stakeholders can understand and verify them. This document will also showcase enough details so that developers can design and implement TasweetX.

This document focuses strictly on the system requirements and components rather than other components of the project such as feasibility and cost-benefit analysis. That is outlined in the feasibility document submitted earlier this semester.

TasweetX is a voting application based on the blockchain domain. It utilizes blockchains immutable features as well as a distributed ledger to ensure all votes are accounted for, and that each participant in the network has the exact same copy of the ledger.

## Goals and Objectives

The main goals of TasweetX are:

Allow users to vote for elections without having to visit the voting booth but also ensure that all votes are made by eligible people.

Allow candidates to ensure no votes are removed illegally by an unethical regulatory authority.

Overall, TasweetX should make voting for any election seamless and secure.

## Scope

TasweetX will allow voters to cast votes to their chosen candidates based on the Emirate they are in. Regulatory authorities will be responsible for managing candidates and control the activation and deactivation of the election period. All votes will be anonymous however authentication mechanisms together with smart contracts will ensure all votes are made by eligible human beings.

Because of the smart contract logic, TasweetX will not allow voters to cast more than one vote. The system will also provide a dashboard for voters so that they can see their previous vote and ensure it has a unique identifier which means their vote has been counted.

## Definitions

**Regulatory Authority** - an individual with permission to manage an election.

**Voter** – any individual eligible to vote in an election.

**Smart Contract** – a programmed logic that allows developers to define eligibility of a voter’s participation in an election.

**Ledger** – a log of transactions entered with their timestamps

**Distributed Ledger** – a log of transactions entered with their timestamps that are not hosted exclusively on one machine.

**Candidate** – any party that is nominated in an election and can receive votes.

**Stakeholders** – anyone directly involved with TasweetX during the development phase or after the development phase.

**Use Case** – explains the interactions between the system and users of the system in various scenarios.

**Users** – any individual who directly interacts with the system

**Blockchain** – A distributed network that holds ledgers of transactions that contain data.

**API** – Application Programming Interface

**Encryption –** the process of securing the data using cryptographic functions.

**Authentication –** the process of verifying the identity of an end user.

**Front-end –** the graphical user interface of the system.

**Back-end –** the component of the system responsible for executing different processes of the system

## Document Conventions

TasweetX is still in its early phases. It is natural to see some undecided requirements or features in the system. This document may have sections listed as TBD (to be decided). This means Team F.A.M.S has not decided on a specific decision for that specific section.

## Assumptions

It is assumed that the voter will have proper identification information on hand when accessing the system.

It is assumed that the regulatory authority will verify the authenticity of nominated candidates before registering them on the system.

It is assumed that the voter will have internet access when attempting to access the mobile or web application.

It is assumed that all eligible voters will vote.

## General Design Constraints

## Product Environment

TasweetX will feature 2 main components. A front-end system and a back-end system.

Blockchain is considered a network and database all in one, therefore, the back-end service will comprise of the database which will hold all the voters’ election votes, and the network which will most likely be hosted on a cloud server (TBD) to allow voters to cast votes from anywhere within the country. It is worth mentioning that due to the nature of how blockchain works, the database will also be hosted with the network on the same server.

The frontend will comprise of 1 main service, a web portal. The web portal will be implemented with HTML5 and JavaScript. APIs will be developed to allow the extraction of data from the ledger to be displayed on the web portal.

## User Characteristics

The main users of this system will be Regulatory Authorities & the voters.

We aim to make the TasweetX user interface as simple as possible so that no matter the technical background the users have, it will be easy to navigate and use TasweetX.

Regulatory authorities will be given access to the system through the admin. Once access is provided, regulatory authorities will have options to start and stop an election, add and remove candidates as well as count the total votes for each candidate in their Emirate.

Voters will be required to sign into the system using an authentication-based procedure.

## Mandated Constraints

Currently, we are looking at two blockchain solutions to use for TasweetX. Hyperledger Fabric and Ethereum. Now, Hyperledger Fabric is more feasible as it has all the components of a blockchain system we need to implement TasweetX. Ethereum is considered due to its simple components. However, Ethereum cryptocurrency is required to execute transactions. Security is also a concern with Ethereum.

The user interface will have a web portal which can be accessed from any PC, laptop or mobile phone browser.

## Potential System Evolution

TasweetX currently is being developed for elections hosted in each Emirate. Due to the way Blockchain solutions are built, it will be scalable so implementation in other use-cases will be simple. API’s are already going to be used for all external applications like the web. These API’s can easily be integrated into other systems.

Moreover, TasweetX system will be implemented on a local network for the innovation fair but it can be deployed on a business cloud server so it’s easily scalable.

## Non-functional Requirements

## Usability Requirements

The main stakeholders of the system include the voters, candidates and the Regulatory Authorities. However, only the candidates and the Regulatory Authorities will be a part of the blockchain network as well. The TasweetX system aims to provide an efficient and intuitive interface to its users.

TasweetX’s system will consist of 1 component – a web portal. The web portal will require a desktop/laptop. Moreover, Wi-Fi is necessary for this component to be functional. The web portal aims to be straight-forward as it is only for the voters and candidates to cast/view their votes and these voters could include both technical and non-technical people. For this reason, the GUI for the TasweetX website will be made attractive, interactive and easy to use at the same time. The different buttons and menus when clicked will perform specific functions.

As soon as the website is launched, the layout will be made very simple and will ask the user to enter login details or sign up, in the first instance. In addition, after sign-up/login is complete, there will be a pop-up message that will direct the user and also notify or make the users aware that the personal information they provide when creating/managing their profile, will only be used for authentication mechanisms together with smart contracts purpose. This interface will be made easy to navigate when the voters are creating/managing their profile. For example- one of the selection buttons is ‘Help’, that will show the users on how to use or navigate through the system. There will also be an intuitive layout presented to the voters that will show details about every candidate that stands for the election and this will eventually direct them to select the candidate they wish to vote and then click on the vote option for their vote to be registered/casted. Furthermore, the users will be presented with an option that will allow them to view the dashboard which displays the final total votes for every candidate.

The web portal aims to have an easy to learn and navigate interface for the regulatory authorities since we assume there will be both technical and non-technical people. Although the interface will have a non-complex layout, there will be a lot more buttons and menus to carry out specific tasks, unlike the simple layout that the mobile application will have. For example – there will be an option to create candidates which once clicked will allow the regulatory authority to create/manage profile and at the same time display the option to add the candidate to the election.

In order to create an effective as well as efficient GUI, we will also be conducting surveys on potential users of the system regarding our different proposed interfaces before finalizing the main layouts/interfaces, to measure the user satisfaction goals and meet the user’s requirements and needs.

## Operational Requirements

## Hardware Requirements

TasweetX’s system will consist of 1 component which will be software based – A web portal. This system requires a front-end system and a back-end management system where the blockchain is implemented.

The end users of our system include; the voters who will be able to cast their votes and the candidates who will be able to view/count the number of votes they received. They will be able to do so through the website.

For the back-end management system, we are the development team for the web portal. For this we would require a desktop/laptop with high computational power.

## Software Requirements

TasweetX will require a well-developed interface for both its end users and the backend management system. However, the back-end management system of TasweetX will need to be developed either on Hyperledger Fabric or Ethereum since it is where block chain mechanism will be implemented. A cross-platform of HTML5/Java Script/ CSS will be required to develop the front-end system.

## Performance Requirements

All the components of TasweetX’s system aims to be a quick system with the least number of faults in it. The start-up time of the web portal should not go beyond 10 seconds and should also be able to handle multiple people accessing all at the same time.

The TasweetX voter website will have a user interface that will require the voters and candidates to provide details such as email address, username and password for sign up/login. This data will be stored in the voters’ database for authentication of the user. Thus, the voter website aspires to complete sign up/login within 5 seconds. Once the voters are logged into their account, they will also need to provide personal information in order to create/manage their profile which should not take more than 3 seconds. Since the main purpose of the website for the voters and candidates is to cast/view votes, the process is aimed to be quick with the user switching between menus and buttons that barely takes a second. It will also provide a dashboard for voters to view their votes and this dashboard will be updated every time only after the next system refresh.

Moreover, for the web portal, authentication mechanisms together with smart contracts will ensure all votes are correctly registered and made by eligible human beings. This transaction process should be instantaneous. Since the different regulatory authorities will be responsible for the activation and deactivation of the election period, this process should not take more than 2 seconds.

## Security Requirements

TasweetX broadly categorize for voters, candidates and regulatory authorities. The rights hence vary. Some security features are inherent in blockchain and there are some additional features added.

### Voters:

Anonymity:

Anonymity’s core features are nodes being anonymous and unidentified. The privacy of each and individual voter is protected by the implementation of an effective ring mechanism. The contact between the voter and the ballot list is of high secrecy and not revealed to anyone. A special type of digital signature scheme called ring signature proposed by Ron Rivest, Adi Shamir, and Yael Tauman, allows voters to sign from a group and the verifier cannot identify the identity of the voter. A voter can anonymously sign in the ballot box. To avoid double voting as double spending in blockchain networks, TasweetX adopts single, one-time signature as proposed by Nicolas van Saberhagen. This ensures that voters with one key pair can only vote once.

Voters List:

Each voter is appointed a public key. The public key of every valid voter is appended to the to the voter list after authentication. Authentication of voters can be done decentralized or centralized. If centralized, then it can be done by a third party.

Stealth Address:

While with one-time ring signature, voter can cast vote one time only. But the information of ballot to be made public until the tally phase, the message must be encrypted and send only after the encryption.

So, the voters can cast their vote that will be obscure for the observers. This is done by unlinking payments scheme. This allows senders (here voters) to generate different addresses based on the same public key. In simple terms, we encode voters with the same public key, and they will vote for the same candidate even though they will send transactions to different address. This makes these addresses stealth.

### Candidates:

Each candidate is of either string or any other data type. Stealth addresses must be generated, and then voters must encode it in on elliptic curve. Voters having access to deterministic hash functions, at the tally phase ballots will be counted accurately.

## Safety Requirements

Since the system runs online, it is essential for every voter to be connected to the internet. And the system itself too must not lose internet connection to constantly keep itself updated. If the internet connection is lost in between any time, there can be error in the tallying phase. And if voters do not have internet connection then it is practically impossible for them to vote, especially as our system is available to every individual personal device.

Besides the availability of the internet connection, it is essential for voters to visit the web portal to use the system of TasweetX. Hence, awareness of the interface of interaction available is also crucial.

Blockchain requires a fair amount of computational power to run the system from generating blocks to mine them. While TasweetX aims to be a lightweight voting system, it is assumed to have a fair amount of computational power to run its system.

## Legal Requirements

Team F.A.M.S will have all the copyrights to the TasweetX system. TasweetX is a hypothetical election voting system that will have no real politics involved. Although, this system aspires to be implemented by election committees in different democratic countries/states where voting is as important as every individual’s rights. However, there may be issues with the different laws of different countries that could hinder the real time implementation of our system.

The TasweetX website for voters will ask the user to provide personal information when creating their own profile/account, which may be a privacy issue. In order to avoid this issue, the users will be notified or made aware that the personal information they provide will only be used for authentication mechanisms together with smart contracts that will ensure all votes are correctly registered and made by eligible human beings. Our system will also ensure complete anonymity of the voter. The regulatory authority will also not be exposed to the identity of the voter, except for the fact that the vote is registered/casted by an eligible individual.

## Other Quality Attributes

Time:

It is expected that in ring signature, time spent by voter to sign in the ballot in linear to ring size. Moreover, the generation time and verification time of votes is almost the same and the anonymity is also acceptable.

## Documentation and Training

TasweetX will come with a description of its usage separate for voters, candidates and regulatory authorities. “How to use” as an introductory feature for every user falling in the categories will be available.

For regulatory authorities a separate training manual can also be made available as they will oversee the regulation of running the election.

## External Interface

## User Interface

Two different interfaces will be taken into consideration for TasweetX. One interface for the voters, and one for the regulatory authorities of the emirates.

The interface for the voters will be a website. Features like Signup, Login, Vote and view vote history will be added to the interface. A simple, easy to use yet advanced interface will be designed for the voters that will make it very easy for them to use it and navigate through it.  
The interface for the regulatory authorities will be different from the one for voters. This will include basic functions like login and viewing votes, but in addition to the basic functions. Features like add and candidate, start-stop and election, tallying of votes and managing candidate profiles will be included. This interface will be a web portal to ensure added security for the voting system and prevent any false logins on the authority’s behalf.

## System Features

## Functional Requirements

### Voters & Candidates

|  |  |
| --- | --- |
| **Requirement No.** | 1 |
| **Function** | Create Account Profile |
| Description | Voter creates a new profile for self |
| Inputs | Voter information (name, nationality, id number, age) |
| Source | Keyboard, Device screen |
| Outputs | Profile created |
| Destination | Voter dashboard, voter database |
| Action | Create Voter Profile |
| Requires | Voter Information and system turned on, connected to the internet |
| Pre-condition | Connected to the server and internet |
| Post-condition | Press ‘save profile’ button |
| Side effects | none |

|  |  |
| --- | --- |
| **Requirement No.** | 2 |
| **Function** | Modify Voter Profile |
| Description | Voter changes details of own profile |
| Inputs | Voter Information (updated) |
| Source | Keyboard, device screen |
| Outputs | Profile Updated |
| Destination | Voter Dashboard, voter database |
| Action | Changing existing voter information |
| Requires | Voter to be logged into the system |
| Pre-condition | Should have an existing profile, logged in |
| Post-condition | Press ‘save profile changes’ button |
| Side effects | none |

|  |  |
| --- | --- |
| **Requirement No.** | 3 |
| **Function** | Voter Login |
| Description | Voter signs using username and password |
| Inputs | Username and Password |
| Source | Keyboard, device screen |
| Outputs | Voter connected to server; voter logged in |
| Destination | Voter Dashboard |
| Action | Voter logs in to the system to vote |
| Requires | Voter to enter username and password |
| Pre-condition | Should have an existing profile, connected to the server |
| Post-condition | Press “login’ button |
| Side effects | none |

|  |  |
| --- | --- |
| **Requirement No.** | 4 |
| **Function** | Cast vote |
| Description | Enables user to cast a vote for the desired candidate |
| Inputs | Selection of candidate |
| Source | Device screen |
| Outputs | Vote casted for desired candidate |
| Destination | Voter database |
| Action | Cast vote for selected candidate |
| Requires | Voter to be logged into the system |
| Pre-condition | Voter meets eligibility criteria of smart contract; logged in |
| Post-condition | 1 vote added for the selected candidate as a transaction |
| Side effects | Vote cannot be changed |

|  |  |
| --- | --- |
| **Requirement No.** | 5 |
| **Function** | View vote history |
| Description | Enables user to view vote history (who they voted for and how many votes the candidates have) |
| Inputs | Vote cast; press button to ‘view dashboard |
| Source | Device screen |
| Outputs | Vote history visible to voter |
| Destination | Voter Dashboard |
| Action | View voter dashboard where all information regarding votes, etc is displayed |
| Requires | Voter should have cast their vote, and be logged in |
| Pre-condition | Voter has already cast a vote; logged into the system |
| Post-condition | None |
| Side effects | None |
|  |  |

### Regulatory Authorities:

|  |  |
| --- | --- |
| **Requirement No.** | 6 |
| **Function** | Add Candidate |
| Description | Adds a candidate to the system as a nominee for election |
| Inputs | Candidate details (Name, Nationality, Age, etc) |
| Source | Keyboard |
| Outputs | Candidate added to system |
| Destination | Regulatory Authority Dashboard, TasweetX candidate database |
| Action | Database updated |
| Requires | Fulfill the requirement of the smart contract. After matching with the requirements of the contract, the candidate account is created. |
| Pre-condition | Regulatory Authority logged into the web portal; connected to the internet |
| Post-condition | none |
| Side effects | none |

|  |  |
| --- | --- |
| **Requirement No.** | 7 |
| **Function** | Login |
| Description | Login to the system |
| Inputs | Regulatory Authority’s username and password |
| Source | Keyboard |
| Outputs | Connected to server, View their own dashboard |
| Destination | Regulatory Authority’s dashboard |
| Action | Actions of the authority will depend as per the rights given to the authority. |
| Requires | A verified account of every individual authority to exist. |
| Pre-condition | Pre-existing profile created by admin |
| Post-condition | none |
| Side effects | none |

|  |  |
| --- | --- |
| **Requirement No.** | 8 |
| **Function** | Delete candidate |
| Description | Candidate profile deleted from the system database |
| Inputs | Candidate reference number |
| Source | keyboard |
| Outputs | Candidate profile deleted from system |
| Destination | Candidate dashboard, database |
| Action | Regulatory Authority deletes the information related to the candidate and revokes his rights as a candidate to remove her/his profile. |
| Requires | Regulatory Authority to have the rights and to remove the candidate it must also fulfill the requirements of the smart contract. |
| Pre-condition | Regulatory Authority logged into the system; connected to the server; connected to the internet |
| Post-condition | None |
| Side effects | none |

|  |  |
| --- | --- |
| **Requirement No.** | 9 |
| **Function** | Start Election |
| Description | Election begins allowing voters to cast their vote for their chosen candidate. |
| Inputs | Vote of every voter |
| Source | Regulatory Authority’s device |
| Outputs | Election starts, voters start voting |
| Destination | TasweetX ledger |
| Action | Regulatory Authority begins the election allowing the votes to be stored as transactions into the ledger |
| Requires | Legitimate regulatory authority have rights and fulfill the requirements of the smart contact. |
| Pre-condition | Regulatory Authority logged into the system, connected to server and internet, verified candidates for elections and verified voters |
| Post-condition | None |
| Side effects | Voters will not be allowed to vote before the start |

|  |  |
| --- | --- |
| **Requirement No.** | 10 |
| **Function** | End Election |
| Description | Voters can no longer vote; the system is turned off |
| Inputs | No input required |
| Source | Regulatory Authority’s device of contact |
| Outputs | Election stops, voters can no longer vote |
| Destination | TasweetX ledger |
| Action | Regulatory Authority stops the election after the pre-decided time to cease the voting process. |
| Requires | Legitimate regulatory authority have rights and fulfill the requirements of the smart contact |
| Pre-condition | Regulatory Authority logged into system, connected to server and internet |
| Post-condition | None |
| Side effects | Any vote after ending election will not be registered into the system |

|  |  |
| --- | --- |
| **Requirement No.** | 11 |
| **Function** | Tally Votes |
| Description | Regulatory Authority tallies votes to see which candidate is getting how many votes (ongoing, as the election is progressing) |
| Inputs | None |
| Source | TasweetX Ledger |
| Outputs | Counting of votes displayed and available to both voters and candidates of their respective parties to ensure the legitimacy |
| Destination | Dashboard of voters, and regulatory authority, and TasweetX Ledger |
| Action | Regulatory Authority checks the vote count |
| Requires | Regulatory Authority to legitimately count votes, it must be in compliance with the smart contract. |
| Pre-condition | Voting must be ongoing |
| Post-condition | No. of votes of each candidate displayed to voters |
| Side effects | none |

|  |  |
| --- | --- |
| **Requirement No.** | 12 |
| **Function** | Announce Results |
| Description | After the election is over, voting stops, no voter can vote after this and Regulatory Authority after tallying the votes announces the results of the election publicly available to both voters and candidates to verify the legitimacy of the result. |
| Inputs | None |
| Source | TasweetX Ledger |
| Outputs | Result of the election |
| Destination | Dashboard of voter, and regulatory authorities and TasweetX Legder |
| Action | Regulatory Authority declares the winner of the election |
| Requires | Regulatory Authority must be logged into the system, election must be finished |
| Pre-condition | An election was conducted already with votes cast for each candidate. (It is assumed all the verification of the votes and candidates was done previously). |
| Post-condition | Election is stopped (i.e. voting stage is over). Outcome of the election is released. |
| Side effects | none |

|  |  |
| --- | --- |
| **Requirement No.** | 13 |
| **Function** | Manage Candidate Profile |
| Description | Regulatory Authority manages the profile of the candidate |
| Inputs | Updated information of the candidate |
| Source | Regulatory Authority |
| Outputs | Updated candidate profile |
| Destination | Candidates’ database |
| Action | Regulatory Authority edits/modifies/updates the profile of the candidate |
| Requires | The updateof the profile must meet the requirements of the smart contract |
| Pre-condition | Verified existing candidate and regulatory authority profiles |
| Post-condition | Candidates and voters can view the candidate’s updated profile |
| Side effects | Time consuming: to manage the profile of every candidate will be taxing on time as the number of elections running increase. |

# Design

## Introduction

TasweetX is a blockchain-based election voting system that ensures all votes are accounted for and can not be tampered with. This document defines the design aspects of the system in line with the requirements identified from the Requirements Document. This document will also showcase the design workflow and plan developers can utilize to implement TasweetX.

## System Overview

TasweetX is being developed to ensure voting elections are fraud-free and that all votes are accounted for. The system will consist of 3 main subsystems:

### Web Application

Regulatory Authorities, candidates, and voters can access the electronic ballot via the website. The website will feature all the functionalities of the system and will be identified as the main dashboard of the entire system.

### Core Blockchain

Using Hyperledger Fabric, the core of the system such as the ledger itself, smart contracts and other logic will be implemented.

## Design Map

This software design document outlines both the high level design and low level design of the TasweetX system.

The design document will focus primarily on the following aspects of the TasweetX system:

The flow of the system as a whole

The Hyperledger Fabric network

The user interfaces for both end user groups (ie. Voters & Authorities)

## Definitions and Acronyms

**Voter** - any individual eligible to vote in an election.

**Smart Contract** - a programmed logic that allows developers to define eligibility of a voter’s participation in an election.

**Ledger** - a log of transactions entered with their timestamps.

**Distributed Ledger** - a log of transactions entered with their timestamps that are not hosted exclusively on one machine.

**Candidate** - any party that is nominated in an election and can receive votes.

**Stakeholders** - anyone directly involved in TasweetX during the development phase or after the development phase.

**Use case** - explains the interactions between the system and users of the system in various scenarios.

**Users** - any individual who directly interacts with the system.

**Blockchain** - A distributed network that holds ledgers of transactions that contain data.

**API** - Application Programming Interface

**Encryption** - the process of securing the data using cryptographic functions.

**Authentication** - The process of verifying the identity of an end-user.

**Front-end** - the graphical user interface of the system.

**Back-end** - the component of the system responsible for executing different processes o the system.

**Hashing** - the process of converting data securely for transmission.

**Endpoint** - the point between where data is being transmitted and where data is being received from that transmission.

**Off-Chain** - processes, logic, or execution of functions that happen outside the blockchain network.

**On-Chain** - processes, logic, or execution of functions that happen within the blockchain network.

## Design Assumptions

### Assumptions

TasweetX is entirely online and can be accessed from an internet connection. We assume the user has a stable internet connection and possesses a modern personal computer or a laptop.

We also assume that the blockchain network (ie the core) is secured. However, the endpoints from the blockchain network to the mobile or web applications are something of a concern. If not secured, anyone can intercept the data (ie the vote) and tamper with it before it is entered into the ledger.

### Constraints

TasweetX is a blockchain application for elections being developed for the 3 main Emirates in the United Arab Emirates: Dubai, Sharjah & Abu Dhabi.

This means TasweetX developers would need access to the Emirates ID database in order to allow voters to register on the system using their Emirates ID. Access to the database will also allow for authentication of uploaded Emirates ID to prevent fraud.

Voters will have limited access to the system. They will not be able to interact directly with the blockchain network. Instead, they will be communicating via the web application which will send requests to the REST API and the system SDK.

Regulatory Authorities will have access to their respective Emirates nodes in order to manage candidates in each emirate and toggle election availability.

The main constraint TasweetX developers are faced with is implementing the system in a way that conforms with the UAE policies and standards of elections.

### System Environment

TasweetX core system is developed with Hyperledger Fabric which is a blockchain development tool that is hosted on the network. The user does not need to download any additional software if accessing the system from the web application.

All votes are stored in the blockchain ledger and nothing is stored on the user’s devices.

### Design Methodology

We will be using the Agile-Iterative model to implement TasweetX. We go through the SDLC in iterations until we achieve a system that is functional and meets the requirements outlined in the Requirement Document. Initially, TasweetX will be developed for a small-scale election. After more iterations and testing, TasweetX will be scaled to the final size that suits the use case of this system.

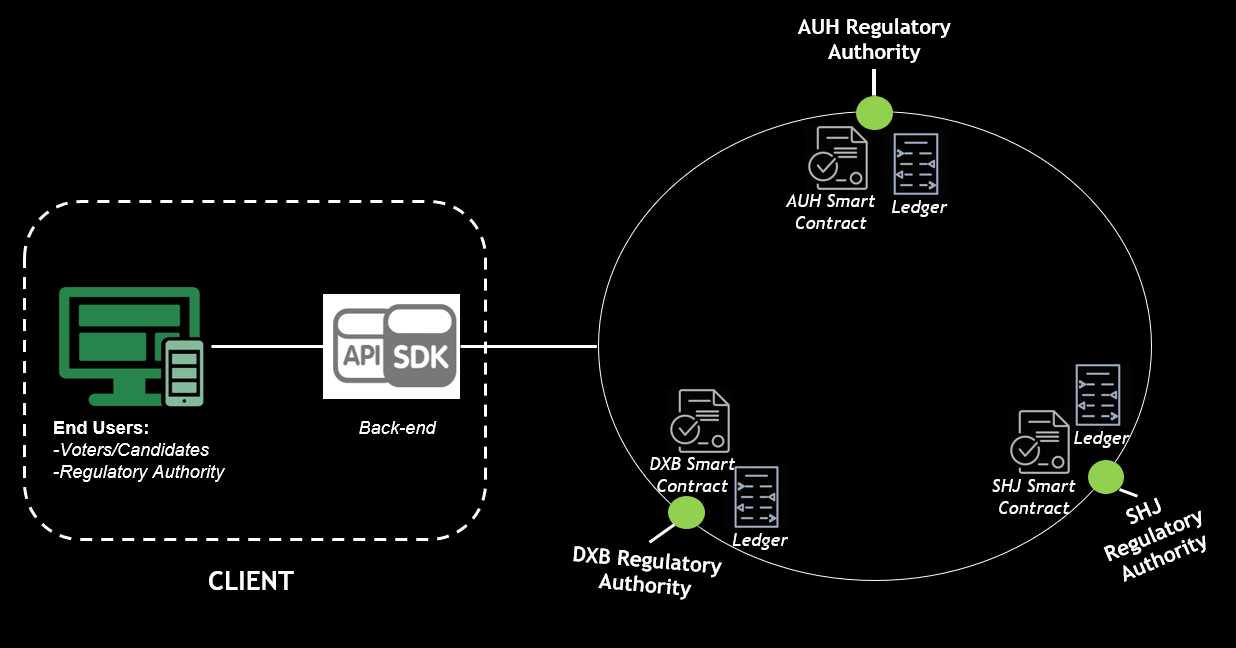
### Risks and Volatile Areas

With only 5 months left on the timeline to deliver the system as promised, the system may still not be suitable for full integration. Developers expect a few issues with the first working prototype of the system and hence will not be confident enough to release the system for live usage.

The UAE is a country that adopts new technology quickly, however the system is based entirely on Blockchain and this is a technology that has not been exposed to regular end users (ie the voters) so adoption by the regular community will be highly risky. The adoption of the system is of highest importance and the developers will ensure the system is as simple and intuitive as possible in order to secure an acceptable adoption rate.

## Architecture

### Overview



The above figure shows the architecture of TasweetX system. TasweetX will allow voters to cast a vote for the desired candidate based on the Emirate they are in. Regulatory Authorities will be responsible for managing candidates and control the activation and deactivation of the election period. All votes will be anonymous however authentication mechanisms together with smart contracts will ensure all votes are made by eligible human beings.

The voter creates an account and using the appropriate API/SDK logs into the system which will then display a dashboard of all candidates standing for election. Once the voter casts a vote for the desired candidate, a transaction request is made where the smart contract then checks the transaction request automatically before adding it to the ledger. Because of the smart contract logic, TasweetX will not allow voters to cast more than one vote. The system will also provide a dashboard for voters so that they can see their previous vote and ensure it has a unique identifier which means their vote has been counted.

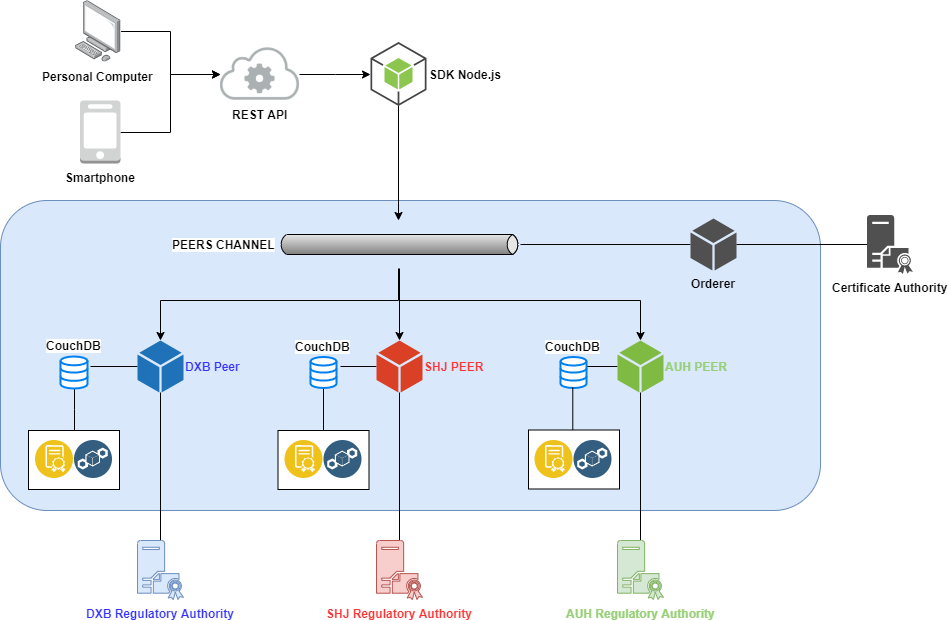
### Rationale

The TasweetX system is a peer-to-peer network. This is because it utilizes blockchain network which consists of immutable features as well as a distributed ledger to ensure all votes are accounted for, and that each participant in the network has the exact same copy of the ledger.

## High-Level Design

### Conceptual View

The below diagram shows the overall design of TasweetX including their components.



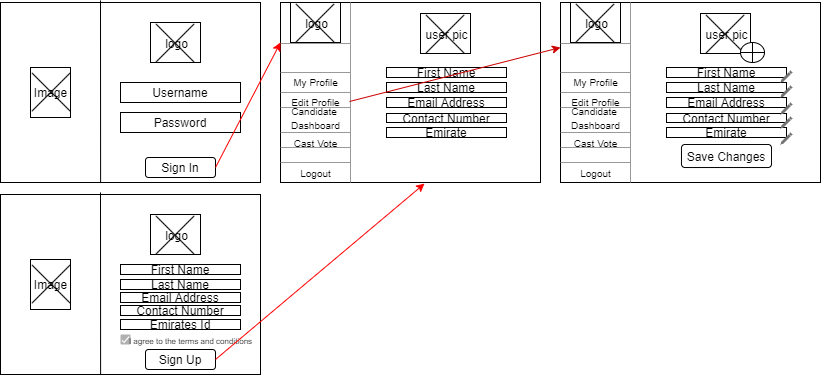
The general workflow of the system is as follows:

* The voter submits a vote request through using a personal computer.
* The request is sent via the REST API to the SDK which will be implemented in the Node.js programming language.
* The SDK processes the request and sends it to the peers channel which is responsible for allowing communication between specific peers.
* All the peers are endorsing peers, which means that the majority of peers need to provide a consensus on whether or not the voter’s vote meets the smart contract requirement and couchDB world state.
* If the majority of peers approve the transaction after simulating the request, the consensus is sent back to the channel.
* If the majority of peers do not approve the transaction, the transaction is becomes invalid and the world state remains unchanged.
* Once the consensus is received in the channel, it is then sent to the orderer which is responsible for ordering the transaction and creating a new block on the chain.
* The vote is then sent to all the peers so that their ledgers remain synchronized across the network and no discrepancies are found.

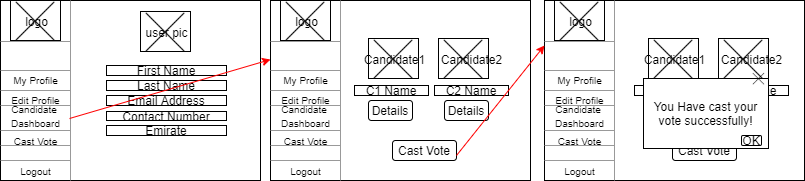
## Wireframes

## Voters

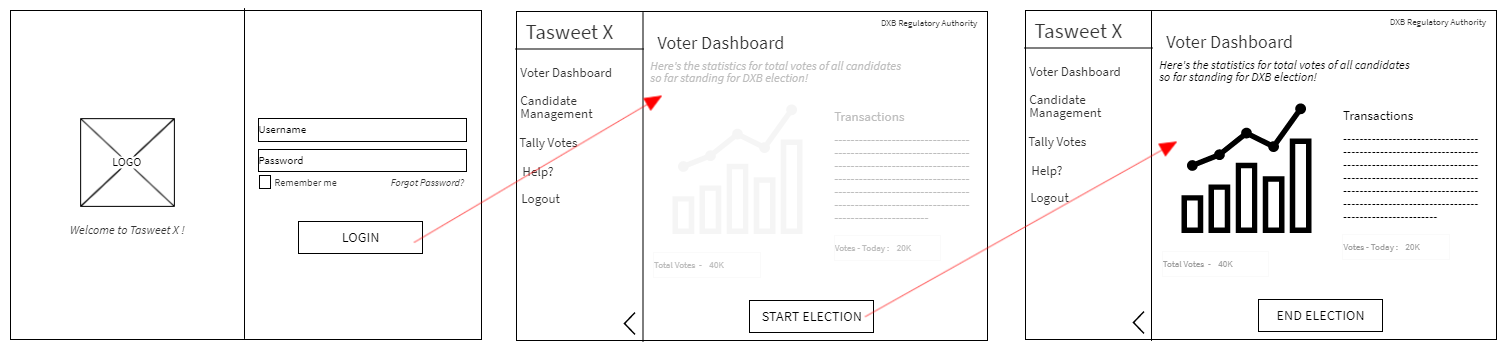
### Sign Up, sign in and view/edit profile



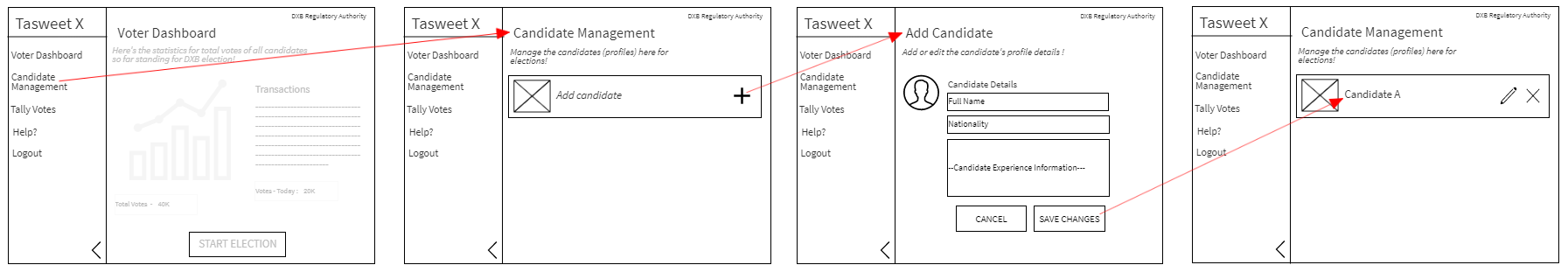
### View Candidate Dashboard and Cast Vote



## Regulatory Authorities

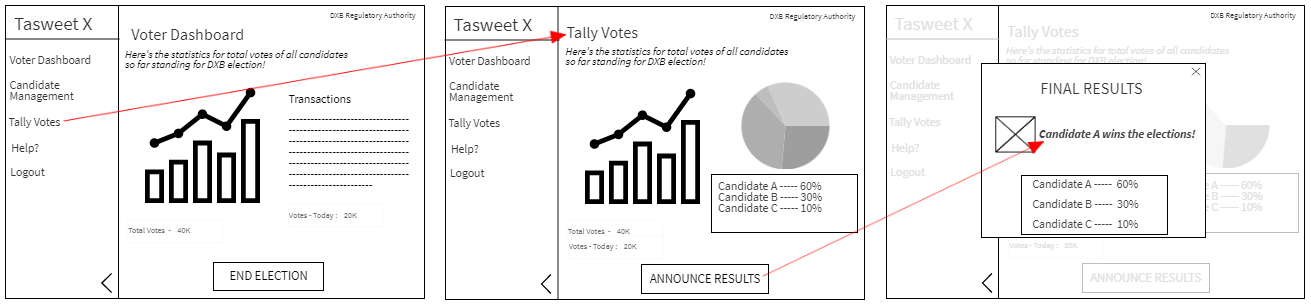
Start/end election (after all candidates standing for elections have been added)

### Add candidate

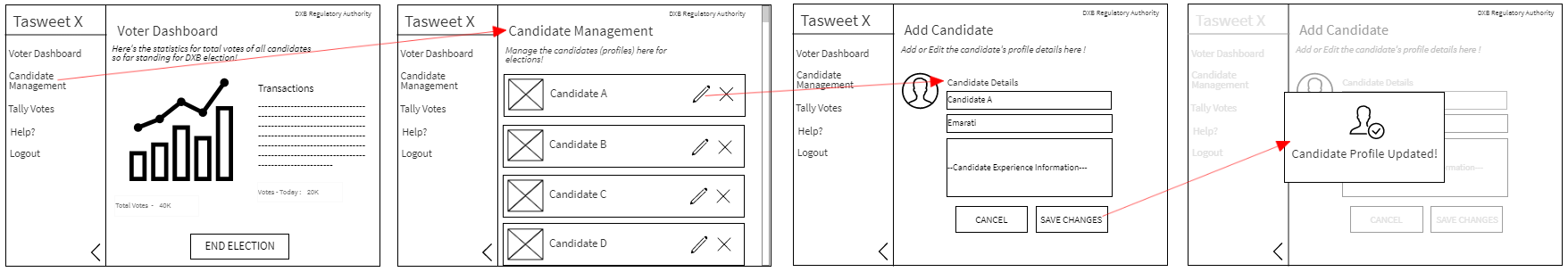


### Delete candidate

### Announce results



### Update Candidate Profile



## Low-Level Design

## Data Dictionary

In this section, we will outline all the databases and their attributes both off-chain & on-chain.

Off-Chain

SDK:

The SDK will be implemented in Node.js and handles all the requests to and from the blockchain network from the web.

|  |  |  |
| --- | --- | --- |
| Variable Name | Data Type | Description |
| username | String | The voters username that will identify them on the system |
| eidNum | String | The voters EID number extracted from their uploaded file when registering. |
| nameFirst | String | The voters first name extracted from their EID uploaded file when registering |
| nameLast | String | The voters last name extracted from their EID uploaded file when registering. |
| dateOfBirth | Date | The voters last name extracted from their EID uploaded file when registering. |
| voteEmirate | String | The emirate the voter is voting in |
| voteID | String | Unique identifier string used to ensure all requests sent to the SDK is unique and not redundant |
| voteCandidateNum | Integer | The candidate the voter is voting for which is identified by pre determined integers. |

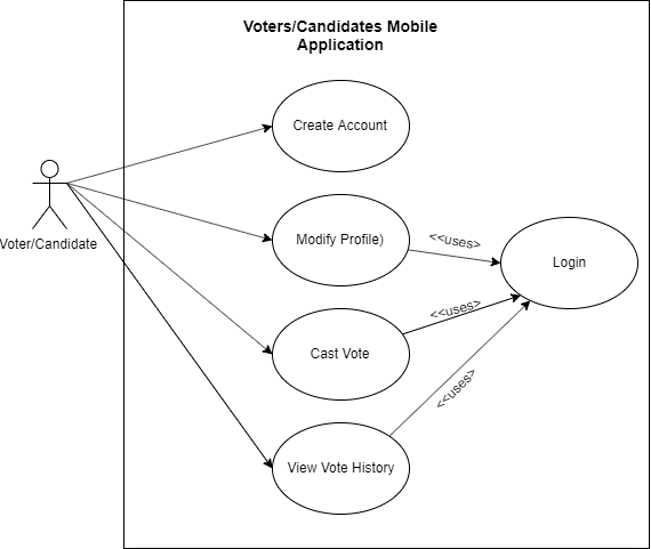
### 

### On-Chain

|  |  |  |
| --- | --- | --- |
| Variable Name | Data Type | Description |
| voterID | String | The voters unique identifier use for identity verification |
| candidateID | String | Predetermined candidate identification number that will be extracted by the SDK for rendering and display on the client applications |
| voteDxb1 | Integer | Number of votes casted in Dubai for candidate 1 |
| voteDxb2 | Integer | Number of votes casted in Dubai for candidate 2 |
| voteShj1 | Integer | Number of votes casted in Sharjah for candidate 1 |
| voteShj2 | Integer | Number of votes casted in Sharjah for candidate 2 |
| voteAuh1 | Integer | Number of votes casted in Sharjah for candidate 1 |
| voteAuh2 | Integer | Number of votes casted in Sharjah for candidate 2 |
| transactionID | String | Unique identifier for the transaction entered into the ledger. |

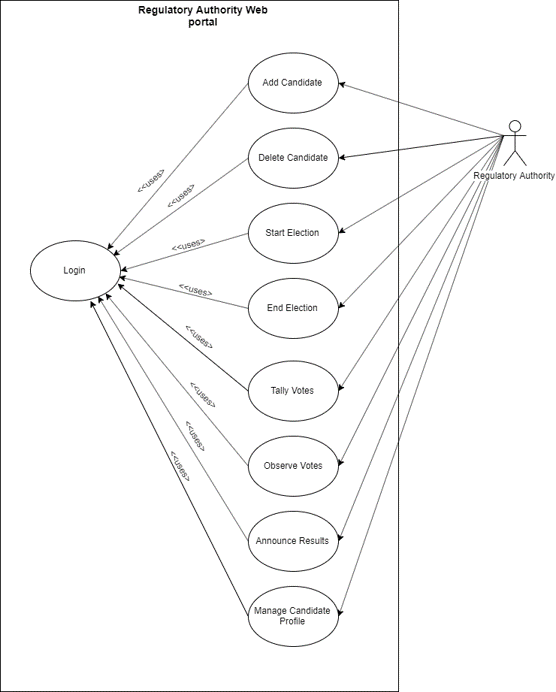
## Use-Case Diagrams

### Voters/Candidates



**Voters/ Candidates Website**

### Regulatory Authorities



Class Diagrams

***Informal Problem Description:***

The interface for TasweetX voting system for voters will open a register page as soon as the website is launched, this page will consist of two options, login and create account. When the create account option is selected, the user will be asked to enter their details in the designated fields and create their new account. If the user decides to go back, there will be a button or the same. If the user is already registered, they can simply login to the system through the login page, by entering their username and password. Once the user is logged into the system, the dashboard consists of various options for the account itself, and for voting. For the account, the user can manage their details and confirm changes. When the voter wants to vote, they can click on the candidate dashboard page where the candidate details of all candidates participating will be available. Once the user selects a candidate another page opens which has the details of the selected candidate and a button to cast and confirm vote. When the voter is done voting, they can view their vote – who they voted for, and statistics of the voting history. Each page has navigation options to go back to the main dashboard, check the help page for help and FAQs, and terms and conditions.

The interface for the regulatory authorities (RA) using the TasweetX voting system will be slightly different from that of the voters. The RAs will be provided with a username and password by the admin which they will use to login to their systems, they will not be allowed to create any new profiles for themselves. Once the RA is logged into the system, they will be provided with options like: Add, Delete and manage candidates, which they can use to add candidate profiles, delete and manage when and if needed for the election. They can start and end elections according to the selected timings. They can also view voting statistics after the election starts, till it ends. Once the election is ended, they can announce results. Additionally, there will be a help page option for the RAs for any kind of help with their Interface design, etc.

***Noun and Verb extraction:***

The interface for TasweetX voting system for voters will open a register page as soon as the website is launched, this page will consist of two options, login and create account. When the create account option is selected, the user will be asked to enter their details in the designated fields and create their new account. If the user decides to go back, there will be a button for the same. If the user is already registered, they can simply login to the system through the login page, by entering their username and password. Once the user is logged into the system, the dashboard consists of various options for the account itself, and for voting. For the account, the user can manage their details and confirm changes. When the voter wants to vote, they can click on the candidate dashboard page where the candidate details of all candidates participating will be available. Once the user selects a candidate another page opens which has the details of the selected candidate and a button to cast and confirm vote. When the voter is done voting, they can view their vote – who they voted for, and statistics of the voting history. Each page has navigation options to go back to the main dashboard, check the help page for help and FAQs, and terms and conditions.

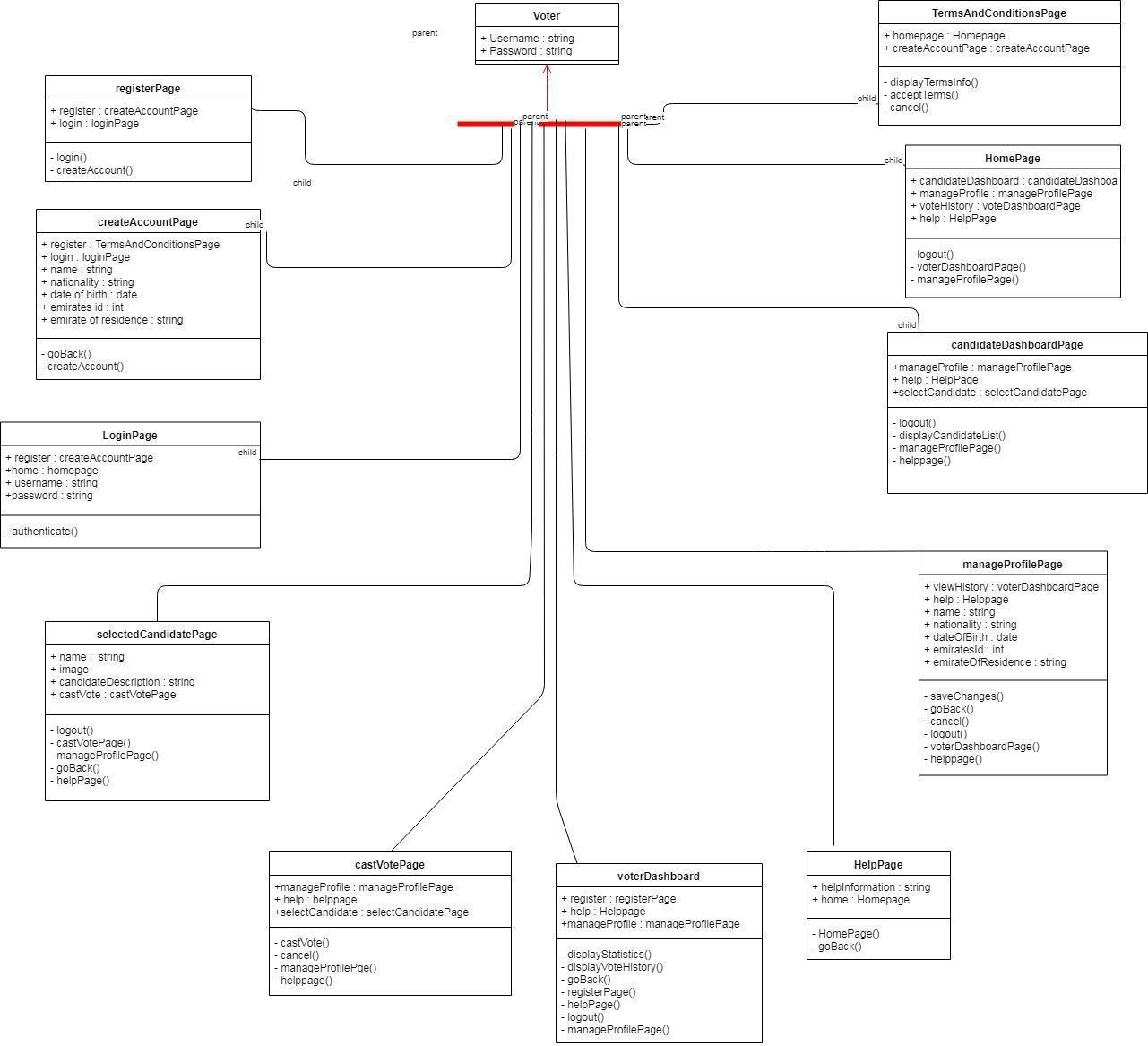
The interface for the regulatory authorities (RA) using the TasweetX voting system will be slightly different from that of the voters. The RAs will be provided with a username and password by the admin which they will use to login to their systems through the login page, they will not be allowed to create any new profiles for themselves. Once the RA is logged into the system, they will be provided with options like: Add, Delete and manage candidates, which they can use to add candidate profiles, delete and manage when and if needed for the election. They can start and end elections according to the selected timings. They can also view voting statistics after the election starts, till it ends on the tally votes page. Once the election is ended, they can announce results. Additionally, there will be a help page option for the RAs for any kind of help with their Interface design, etc.

***Formal Problem Definition:***

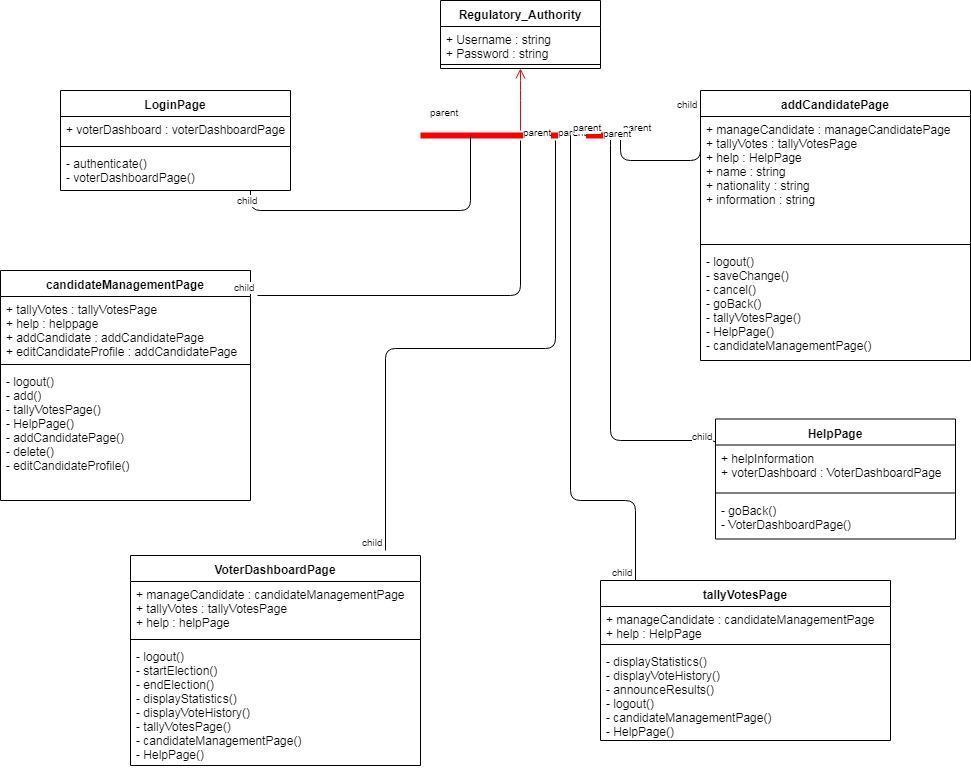
*Nouns*: a register page, login, create account, the login page, the dashboard, candidate dashboard page, selected candidate, statistics of the voting history, dashboard, help page, terms and conditions, login page, Add, Delete and manage candidates, help page

*Verbs*: create account, enter their details, create their new account, go back, login, manage their details, and confirm changes, vote, cast and confirm vote, view their vote , help, login, add, candidate profiles, delete, manage , start, end elections , view voting statistics , announce, results, help

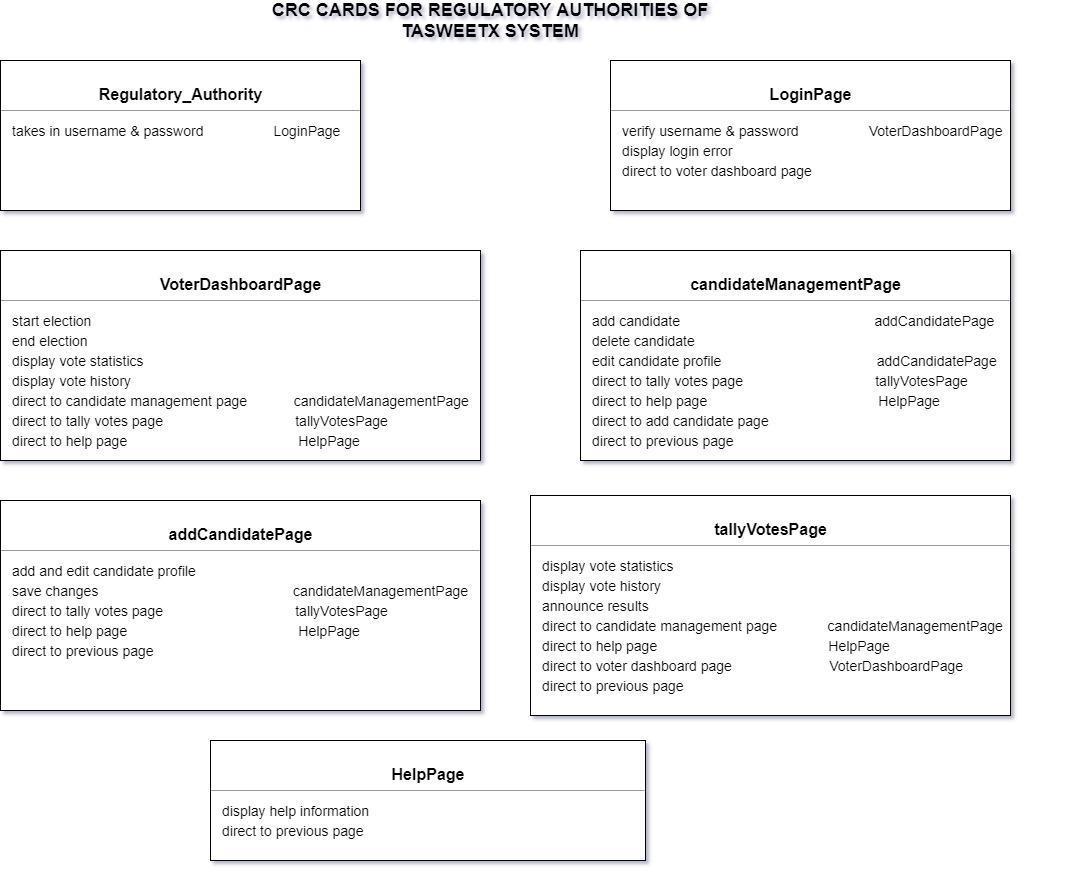
### Voters

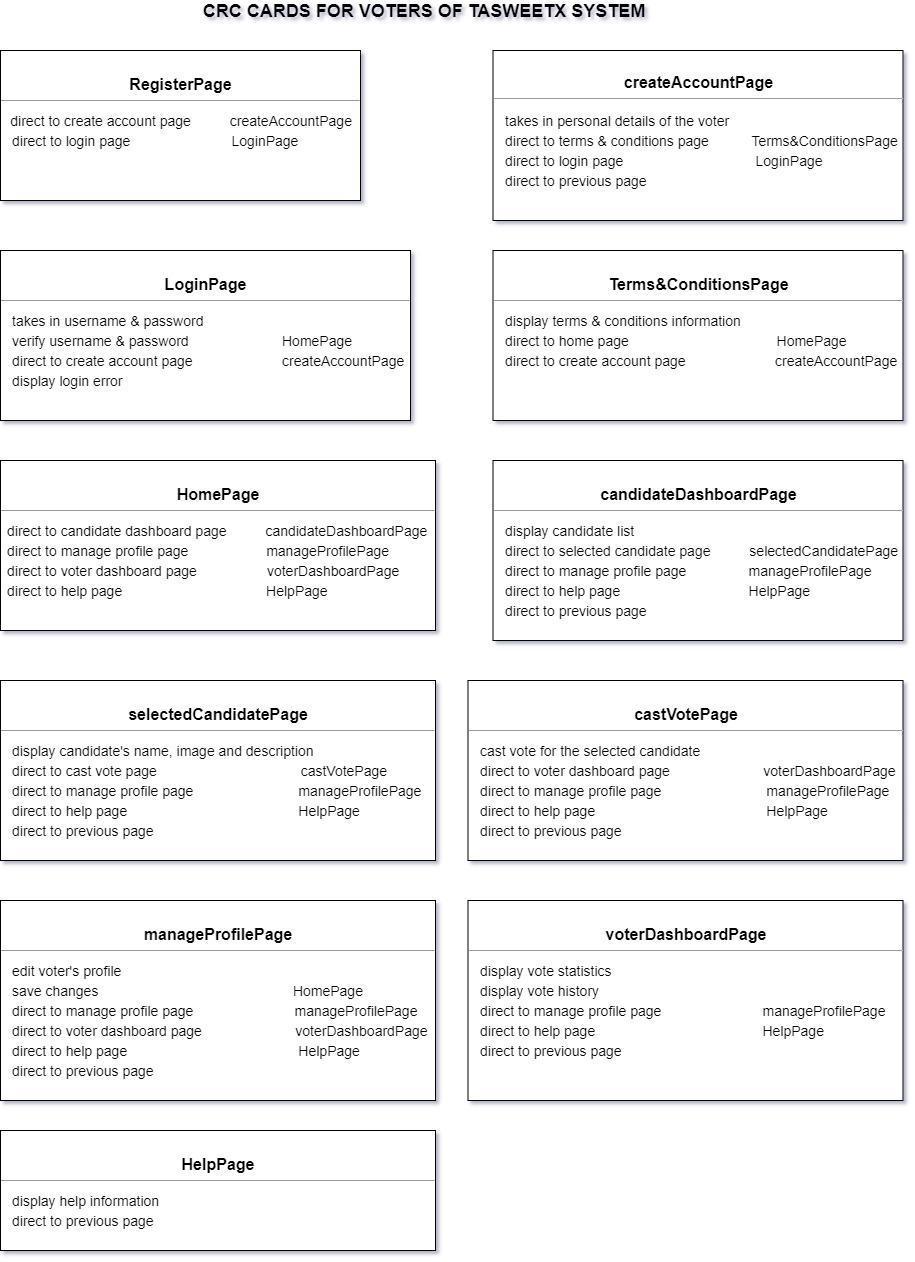


### Regulatory Authorities



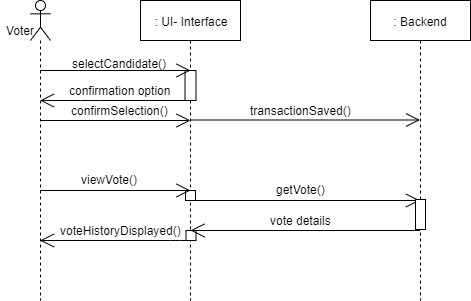
## CRC Cards



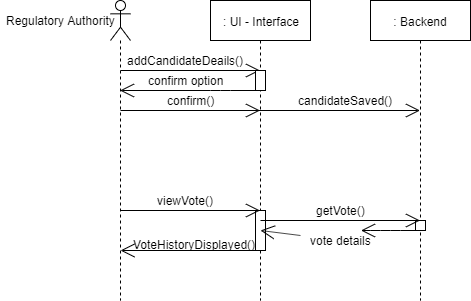


## Sequence Diagrams

### Voter

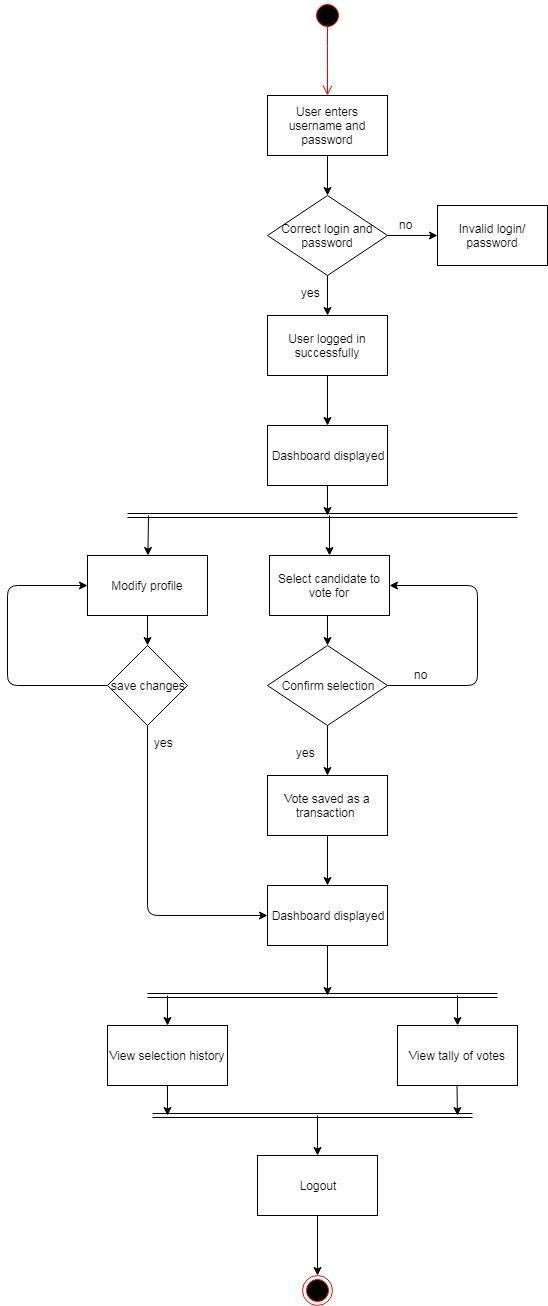


### Regulatory Authority



## Activity Diagrams

Voters

*assuming the voter has already registered with the system.*

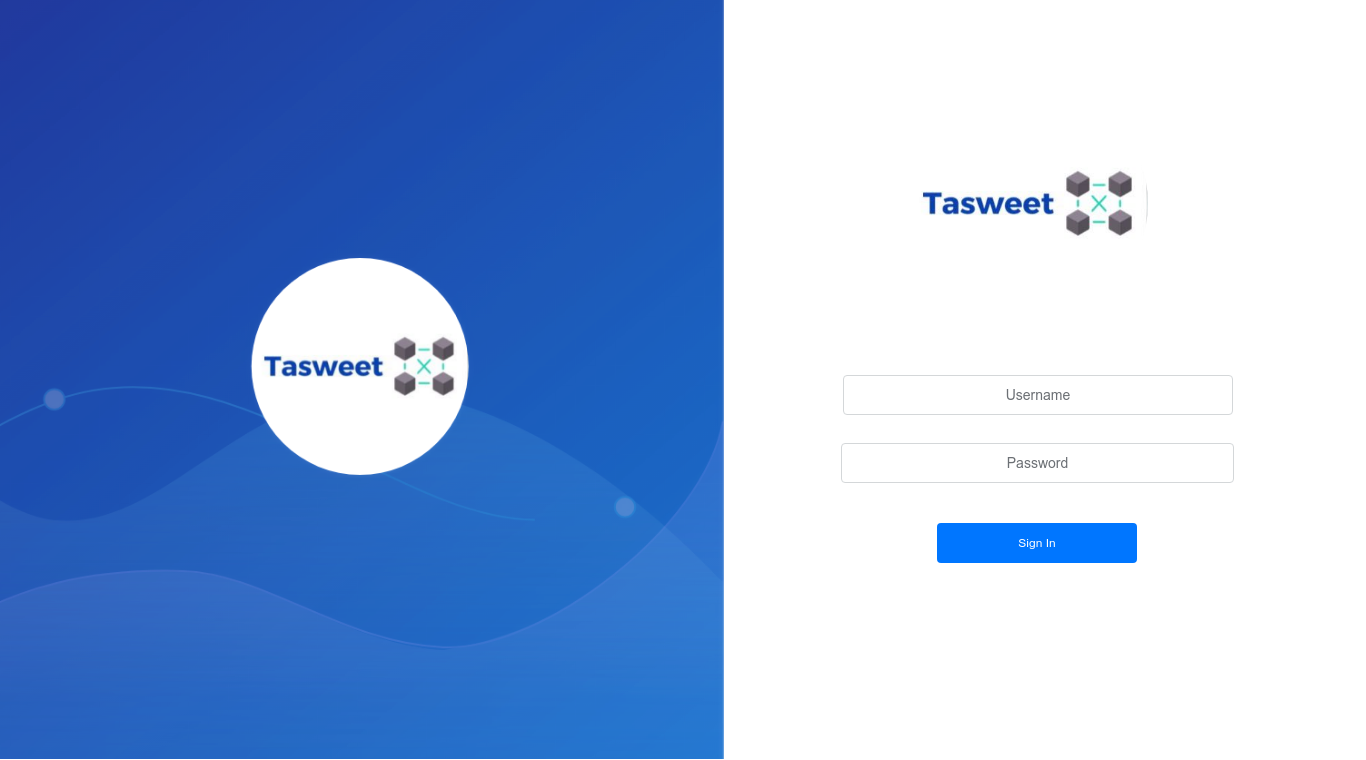
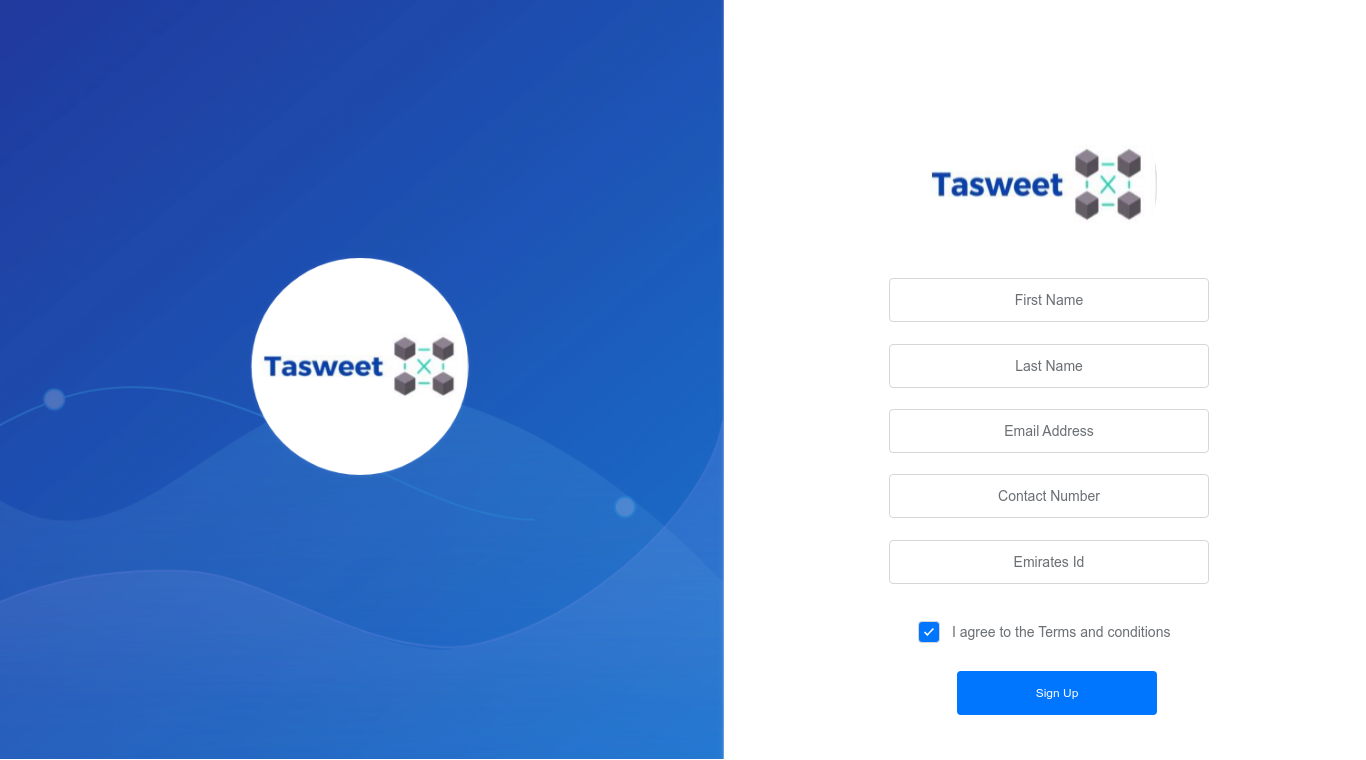
### Regulatory Authorities

## User Interface Design

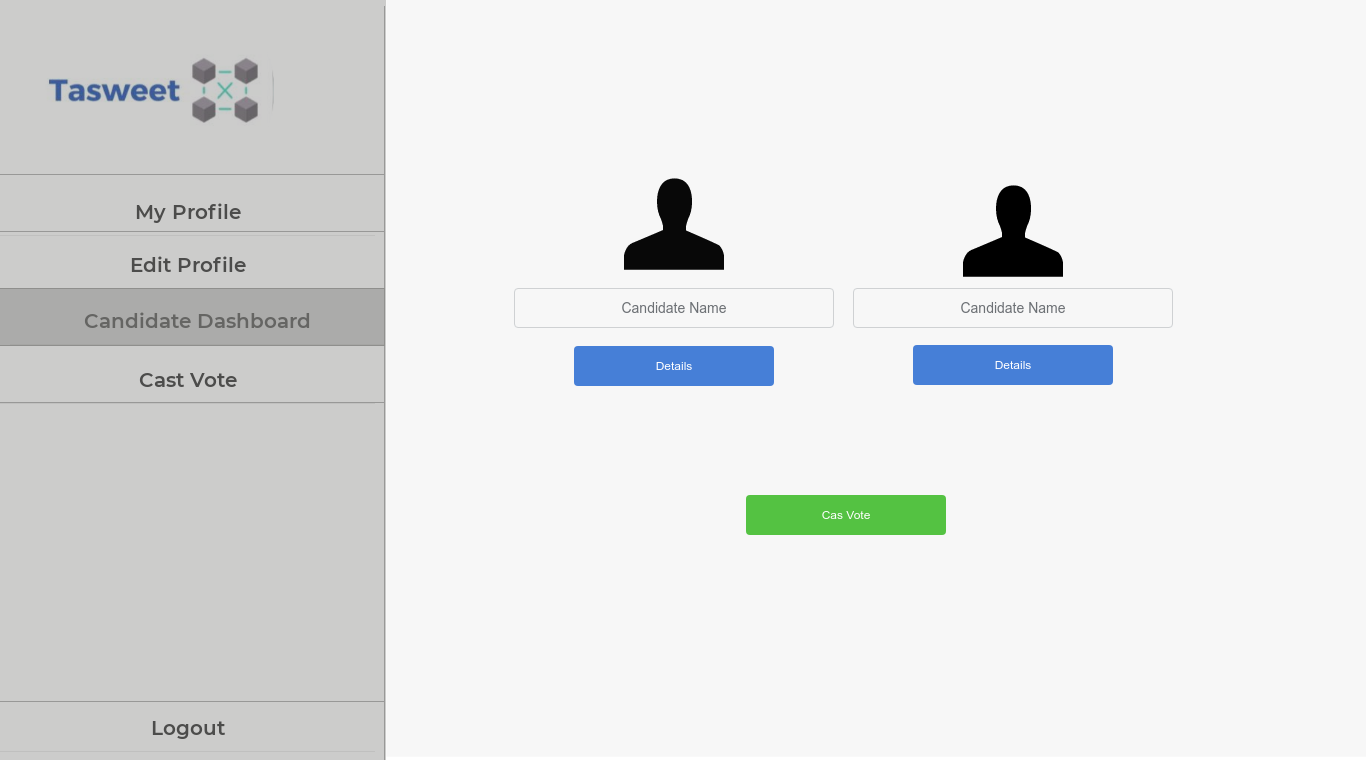
## Website Control (Voter’s Side)

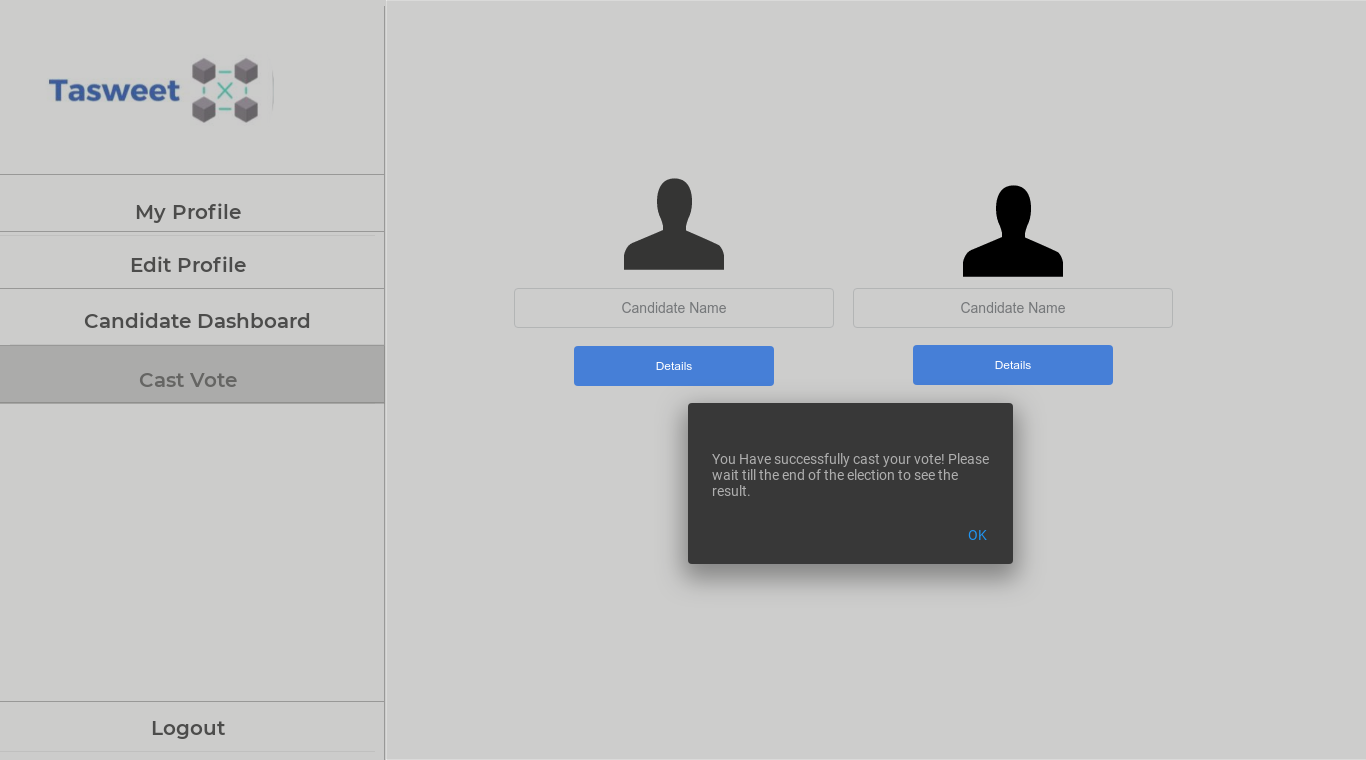
The following screenshots are sample interface design for the website for the voters.

### Sign In & Create Account

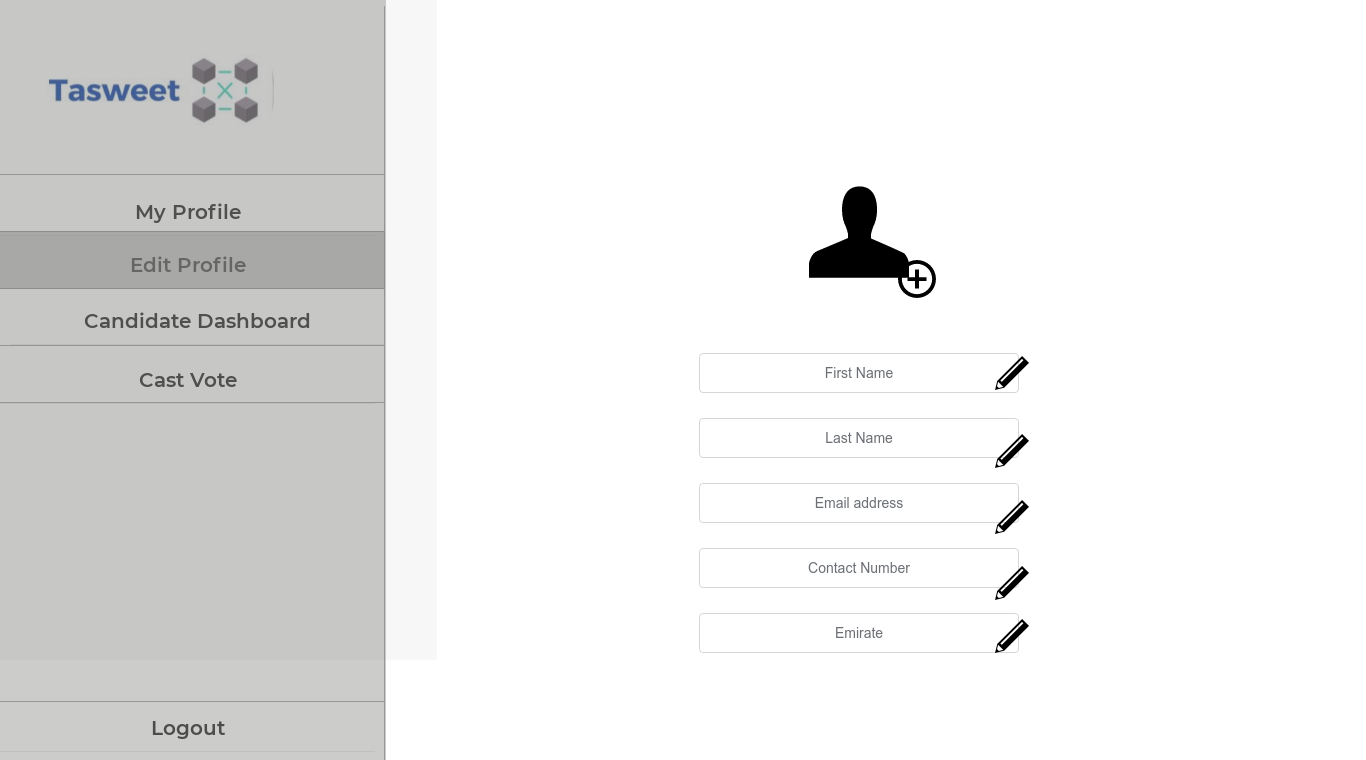
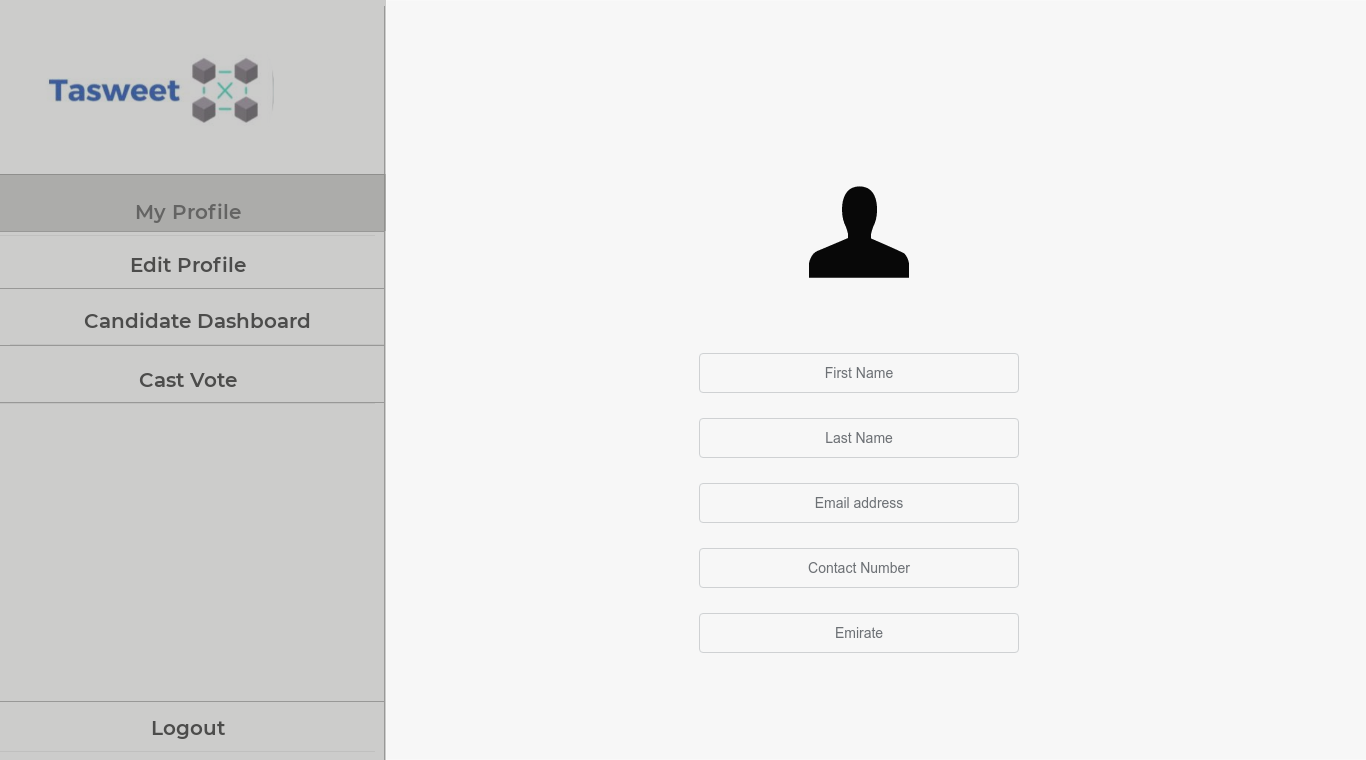


### Candidate Dashboard and Vote Page





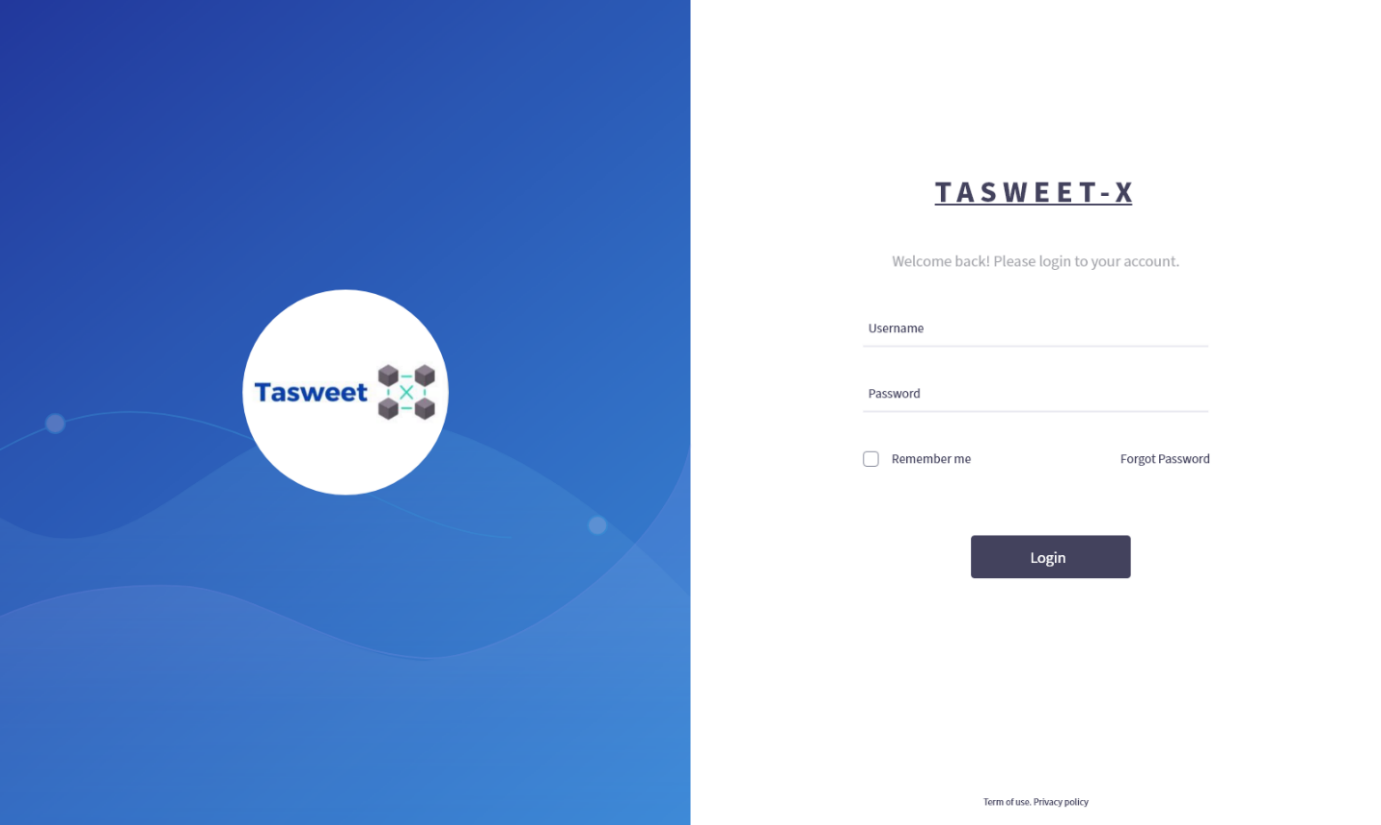
### Voters Profile and Edit Profile



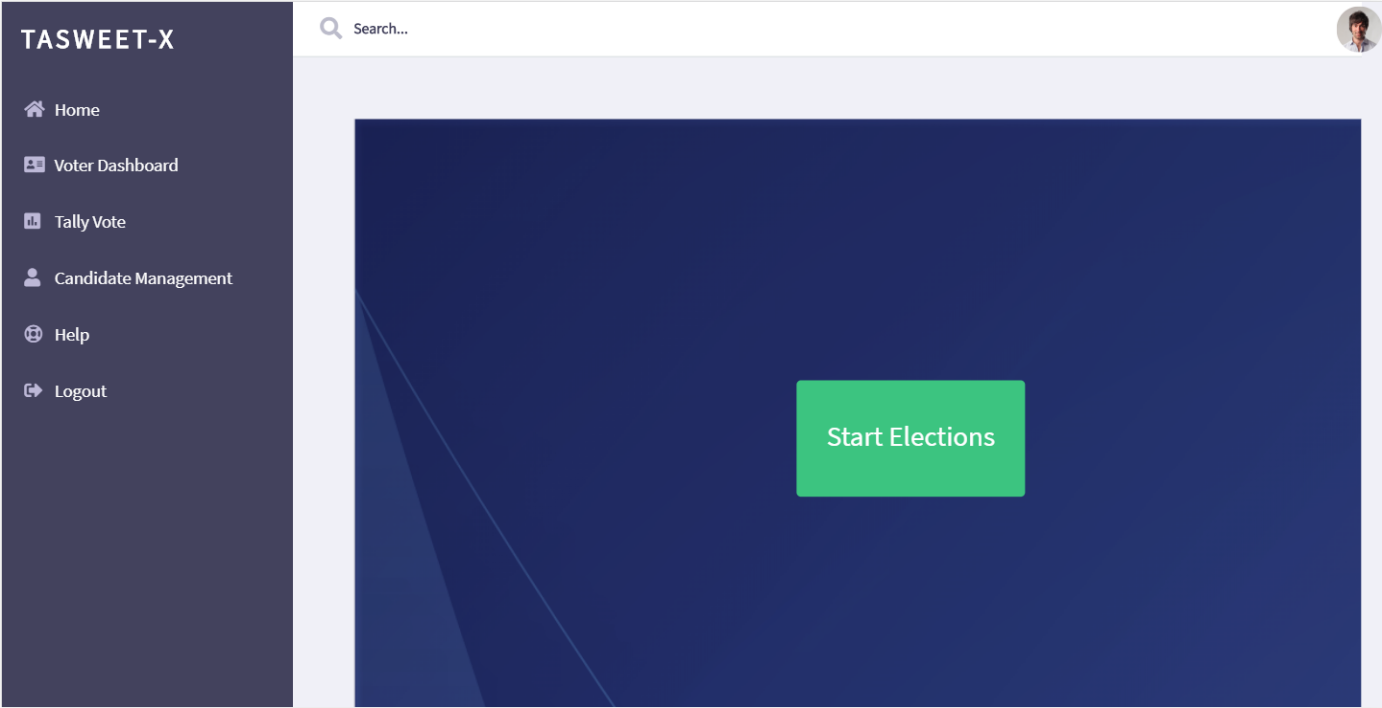
## Web Portal (Regulatory Authority’s Side)

The following screenshots are sample web portal interface design for the regulatory authorities.

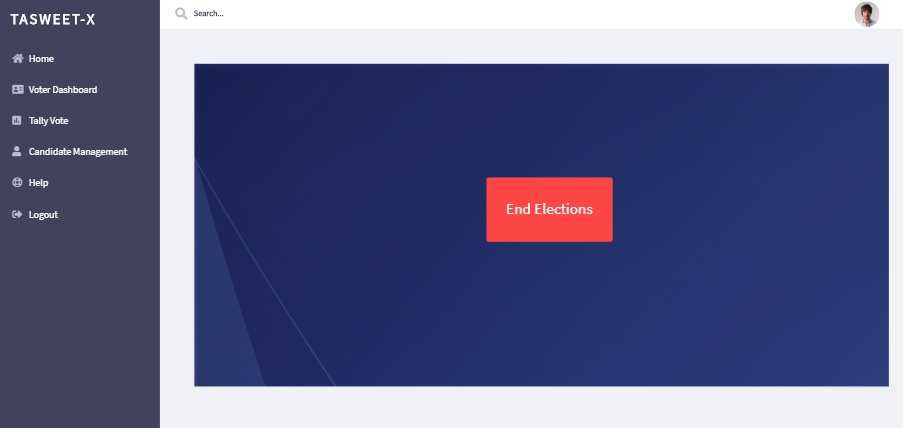
### Log In



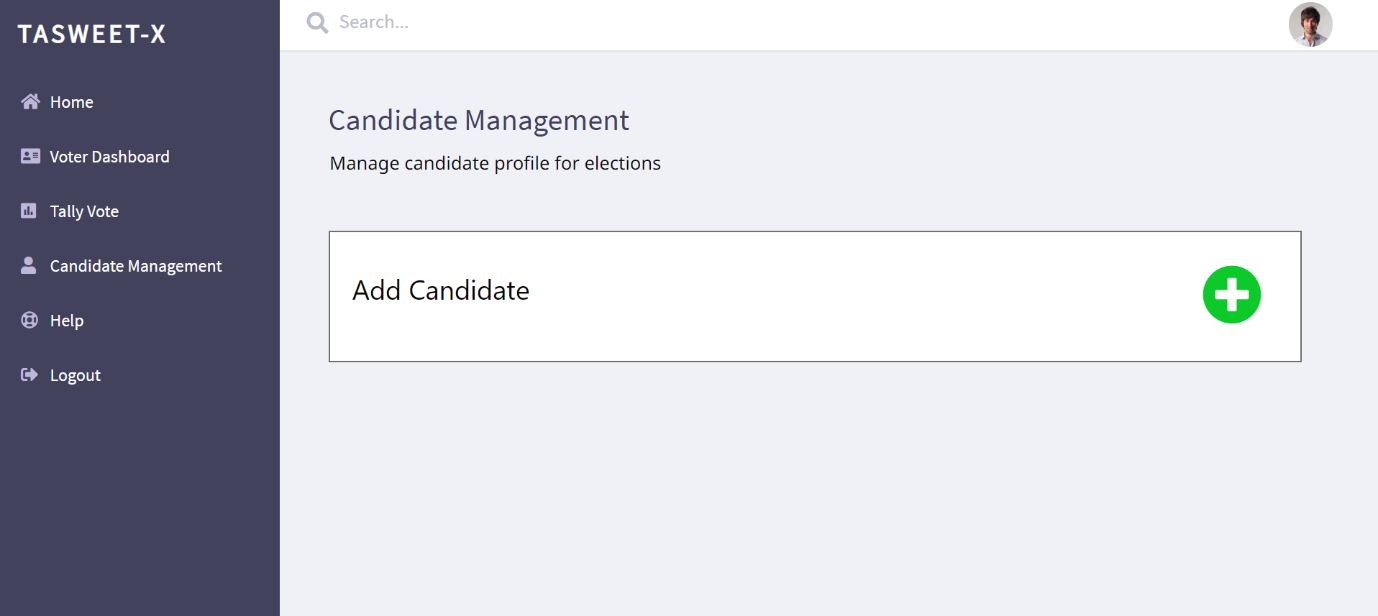
### Start Elections

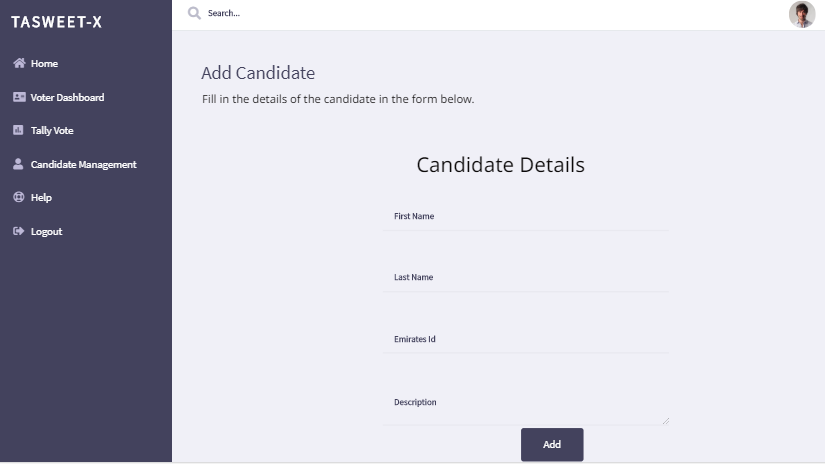


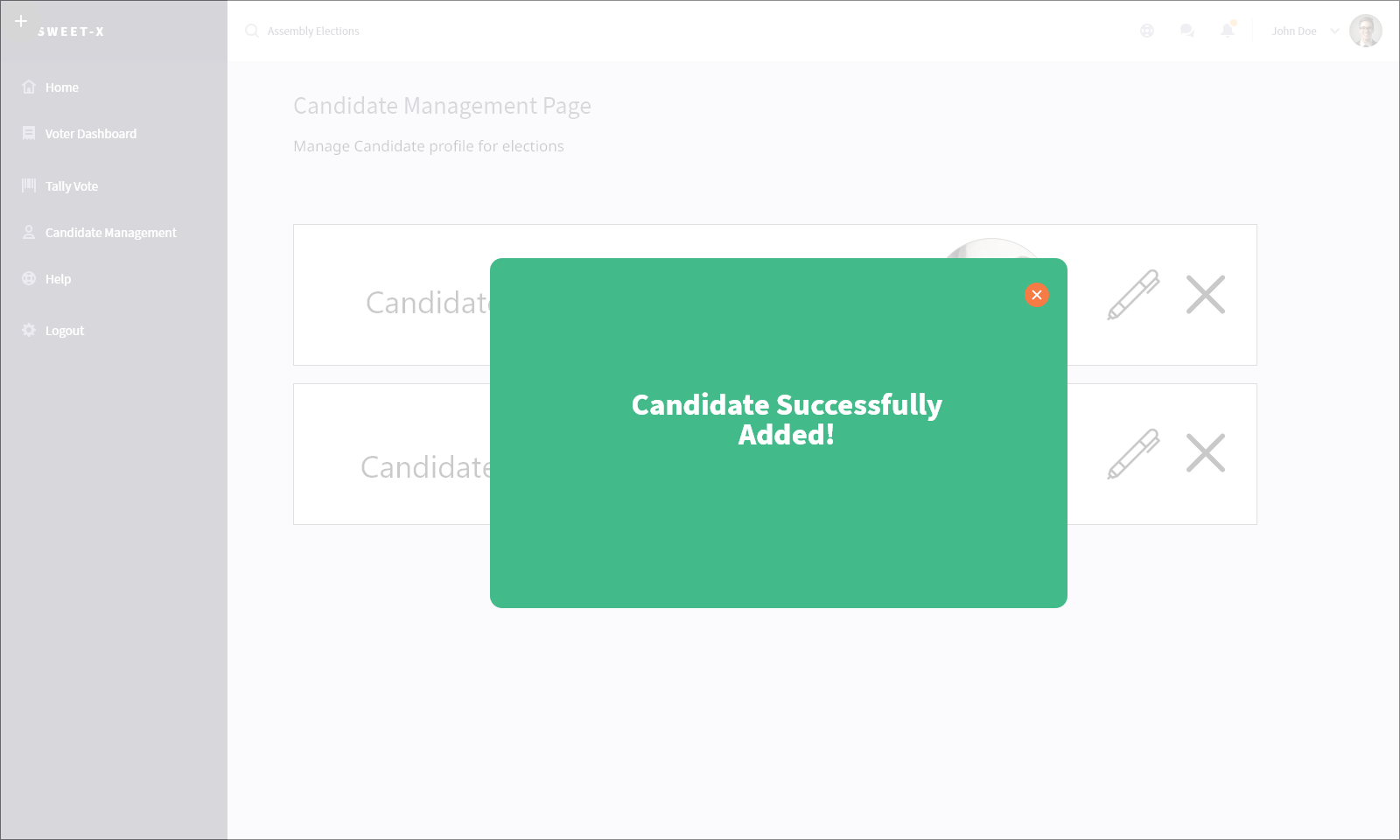
### End Elections

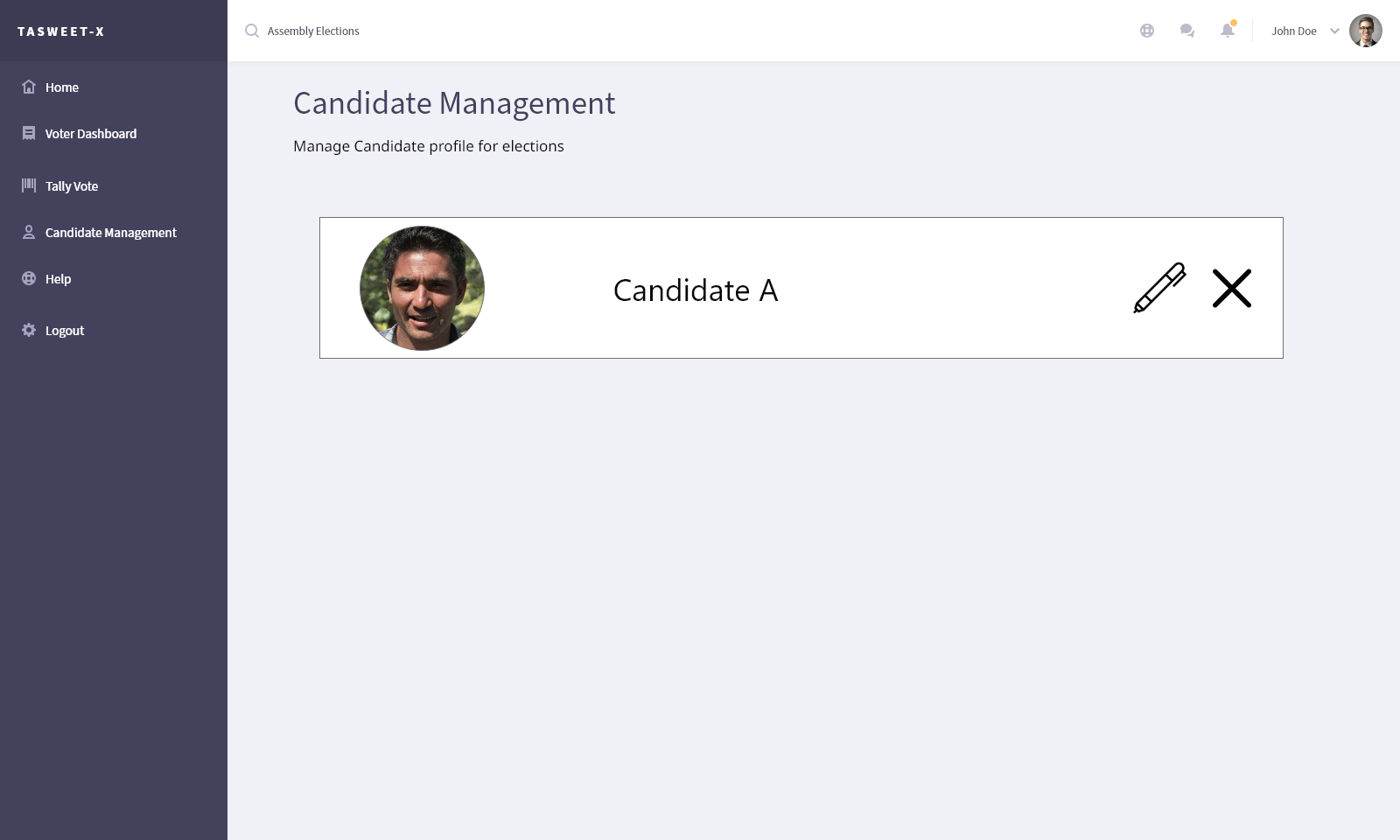


### Add Candidates

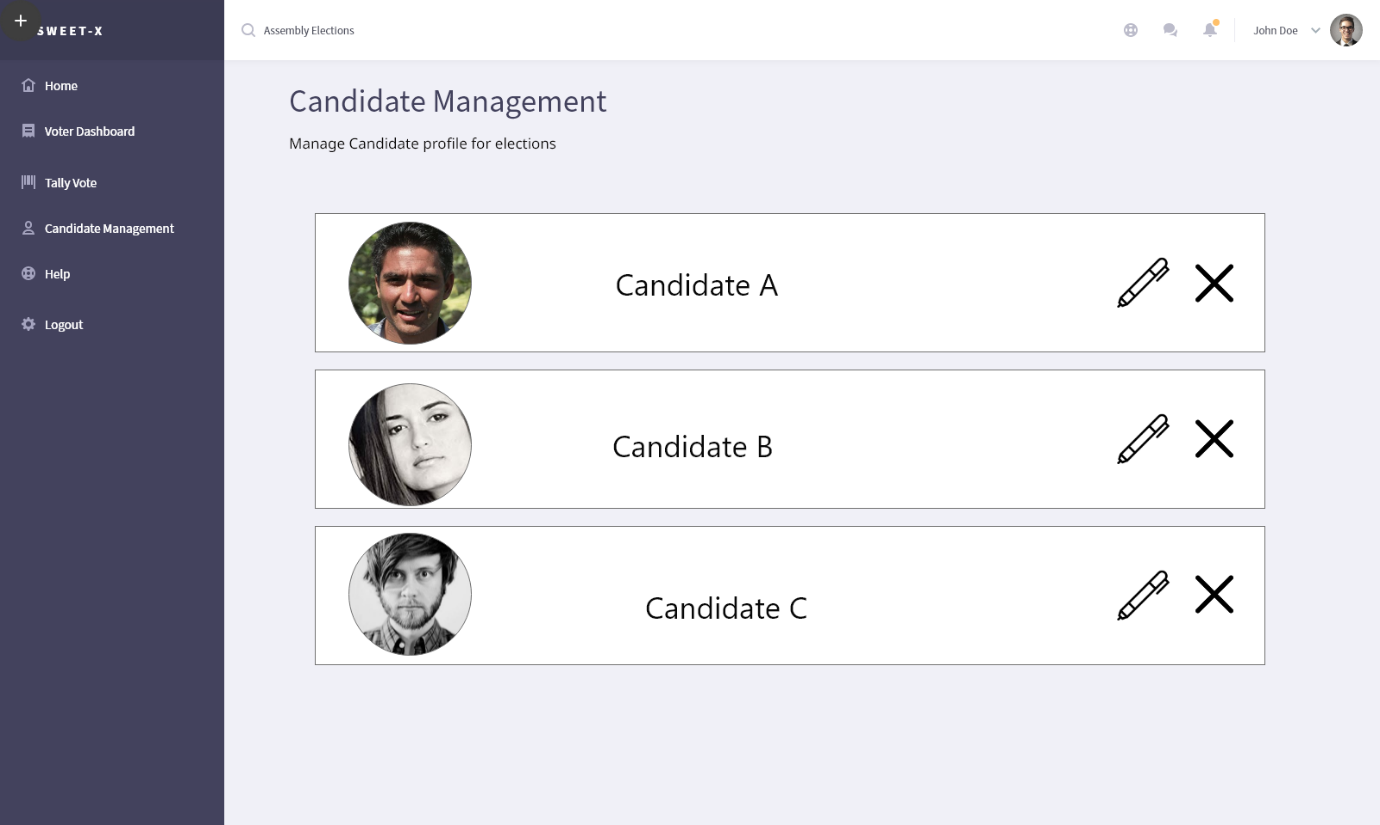


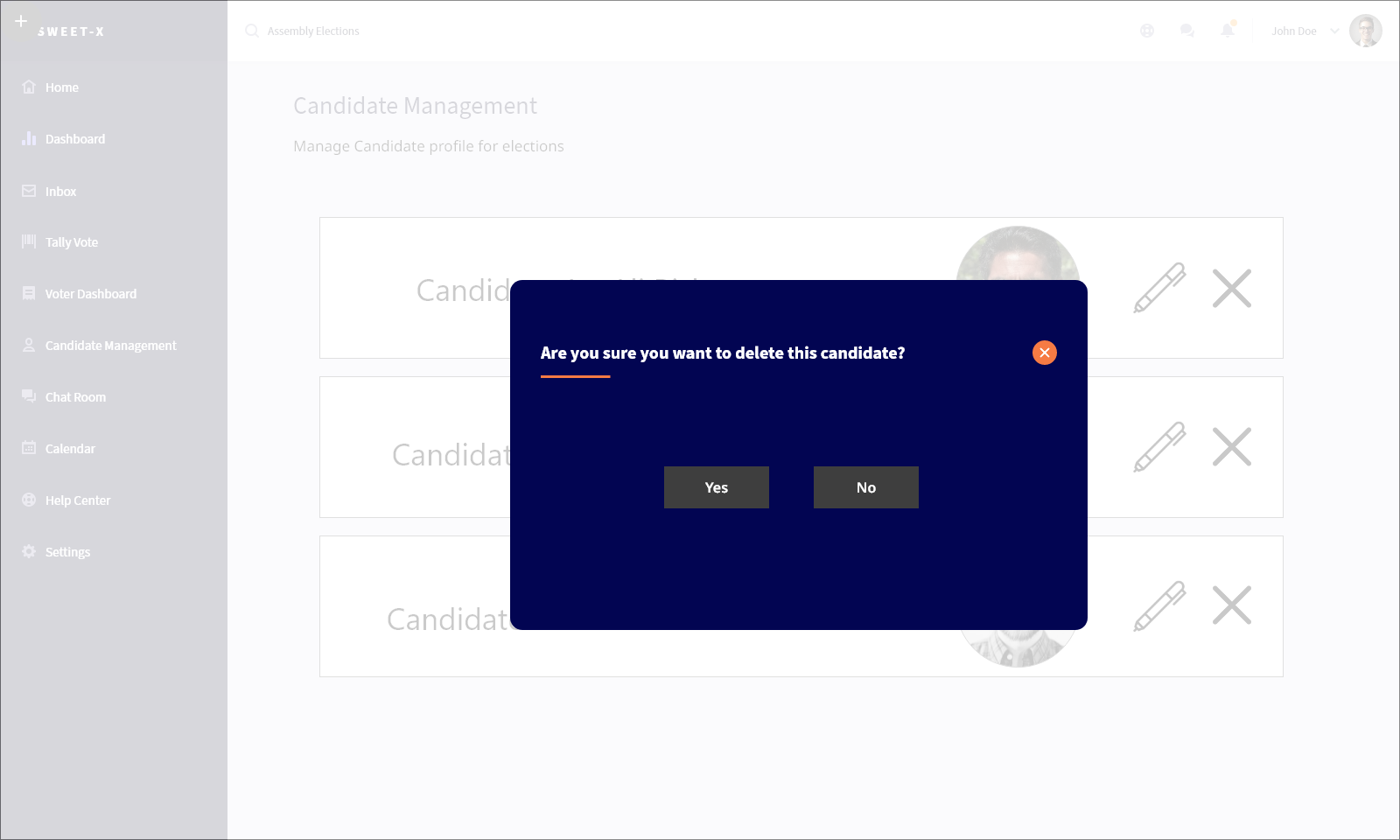




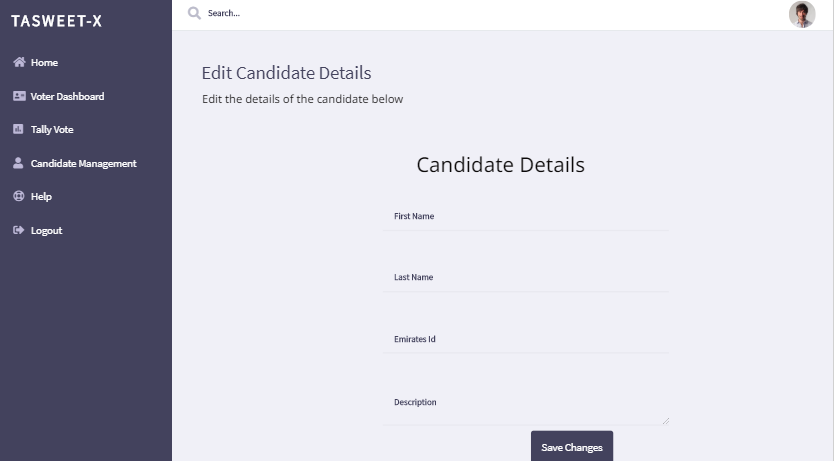


### Delete Candidate

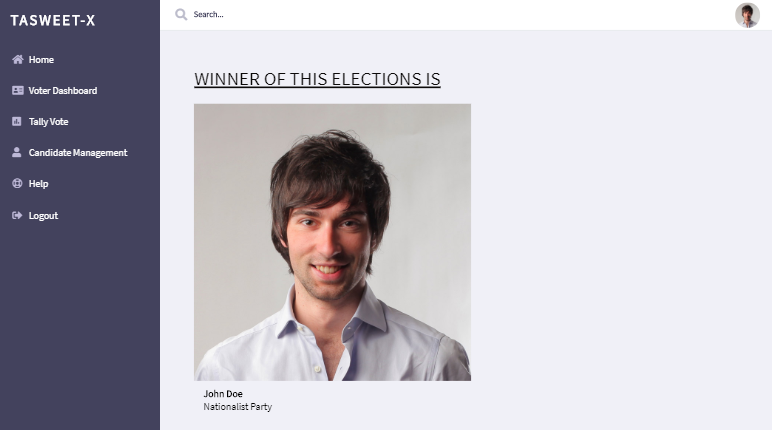
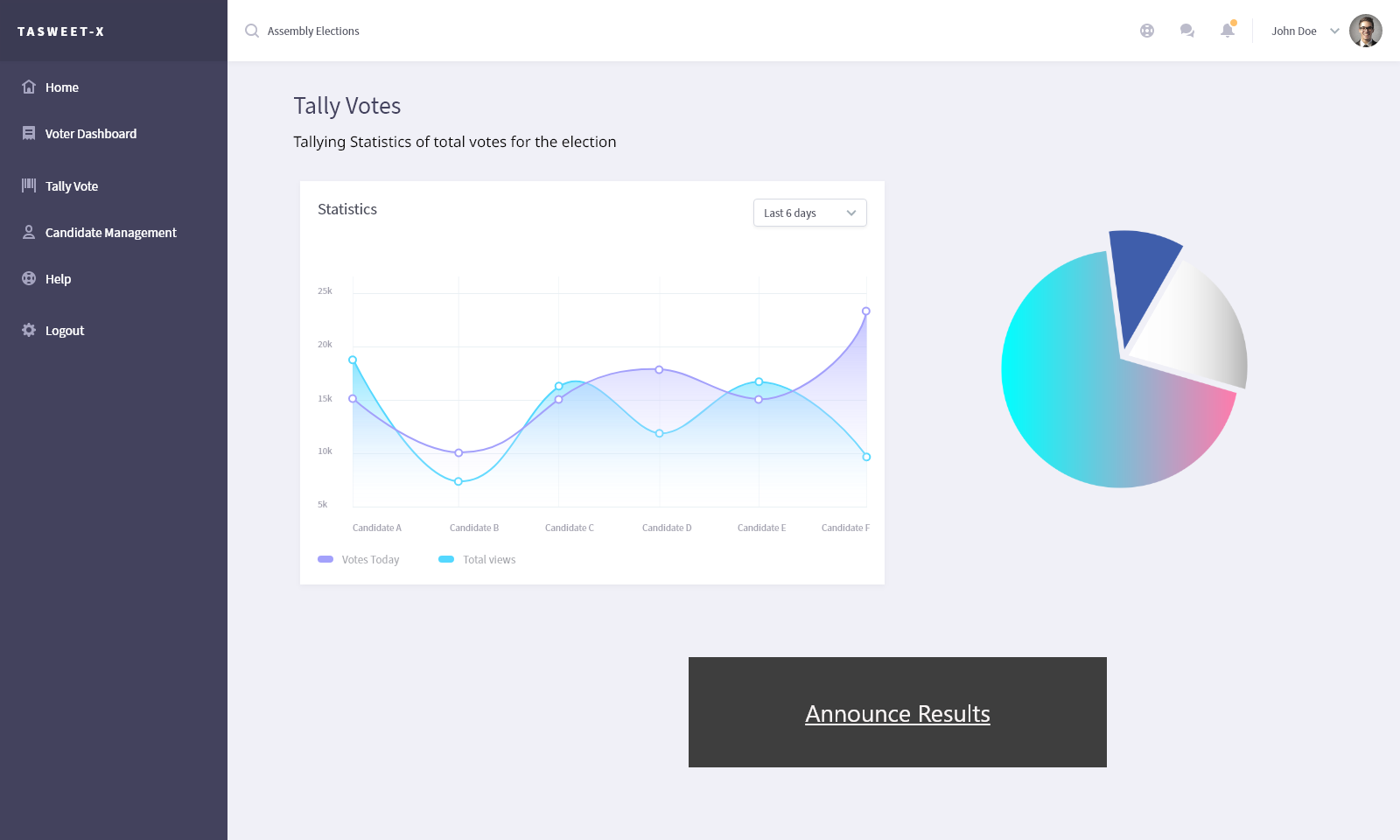




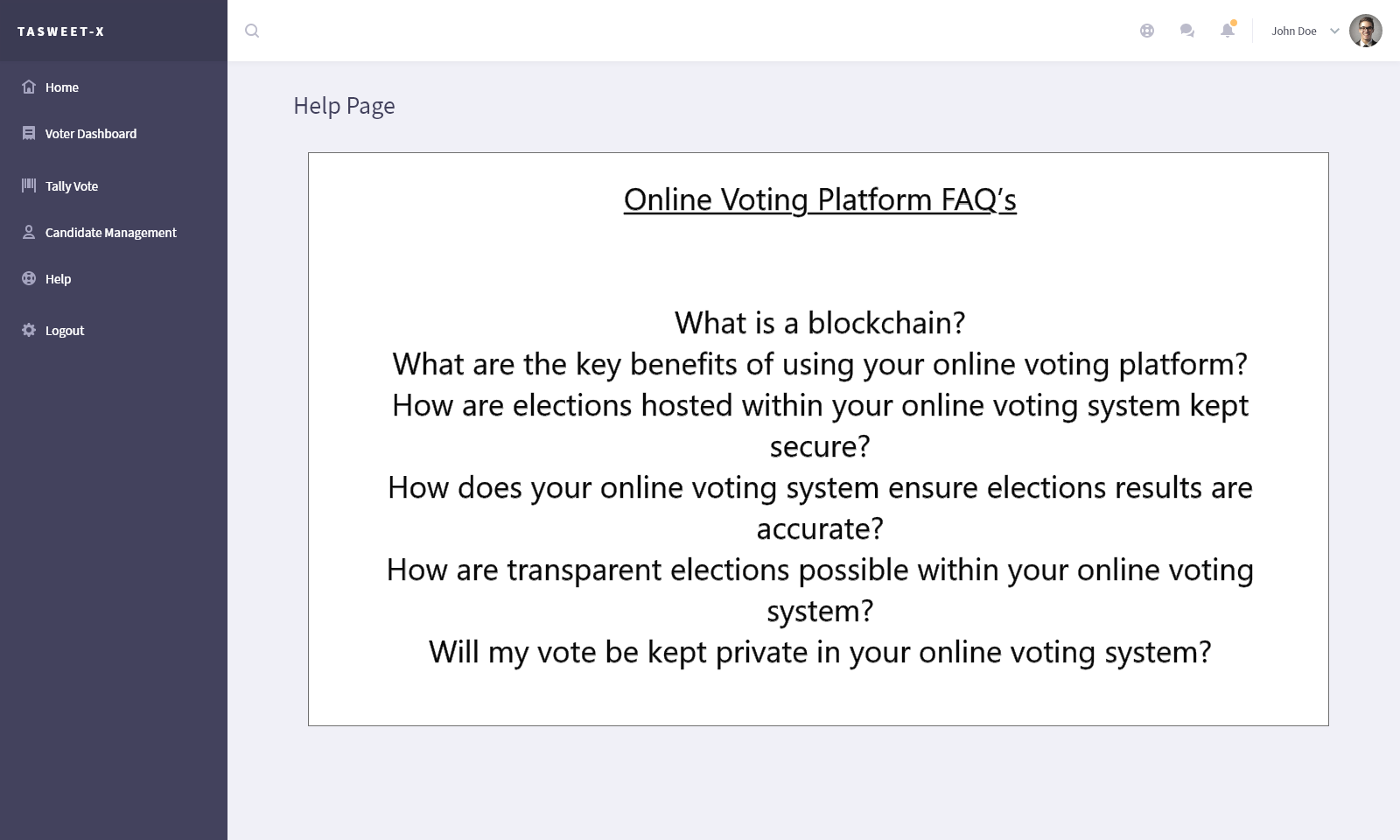
### Edit Candidate Profile



### Announce Results



### Help Page



# Implementation

To begin developing the Blockchain solution using Hyperledger, there are several perquisites required to be installed. Once those perquisites are installed, we can then begin setting up the development environment to have all the tools required readily available.

## Pre-requisites

The Hyperledger framework requires several prerequisites with specific versions to be installed before we can begin installing the development environment. They are listed as follows:

* Ubuntu Linux Operating System – We used an Ubuntu virtual machine from VirtualBox to carry out this project
* Docker Engine @ V17.03+
* Docker Compose @ V1.8+
* Nodejs @ V8.9
* Npm command line tool @ v5.x+
* Git (to be able to pull files from GitHub) @ v2.9+
* Python @ v2.7+
* Visual Studio Code
* Hyperledger Plugin in Visual Studio Code

After the above is installed, we can begin setting up the development environment which will install all the required tools to develop a Blockchain solution on the Hyperledger framework.

## Development Environment

There are a few tools required to set up the Hyperledger development environment

* Composer command line interface @ v0.19
* Composer rest server to generate REST API @ v0.20
* Hyperledger composer generator @ v0.20
* yo command line
* Composer playground was also installed for quick testing purposes @ v0.20

## Installing Hyperledger Fabric

After the prerequisites and the development environment is installed and set up, we now begin to install Hyperledger Fabric locally that we can use to deploy our Blockchain solution to.

We do this by making a curl request to the official Hyperledger read the docs website and then installing it using the scripts inside the file.

## Developing the Blockchain Solution

Using our class diagrams, we must first develop something called a Model File. In this file, all participants, assets and transactions are defined exactly how we need them. The Hyperledger solution uses this file for reference whenever actual coding logic is implemented. A considerable amount of time was spent reworking the model file to ensure that the definitions are not vague and suit the general workflow of the solution. The model file was implemented using the Hyperledger Modelling Language.

After the model file is implemented, we moved on to implementing the smart contract for each possible transaction. All smart contracts were implemented using JavaScript with Hyperledger Assets & Participant Registry.

1. newVoter
   1. This smart contract is called whenever a new voter wishes to sign up. It will grab their details and ensures that their Emirates ID is unique and has not been registered before.
   2. It also automatically sets the “hasVoted” flag to false
2. newCandidate
   1. This smart contract is called whenever a Regulatory Authority wishes to create a new candidate. It will grab all the candidates details and ensures that the candidate ID is unique and has not been registered before.
   2. It also automatically sets the “numVotes” this candidate has to 0.
3. updateVoter
   1. This smart contract allows the voter to update their profiles.
   2. They are only allowed to update their email address, mobile number, and/or their emirate residence
4. updateCandidate
   1. This smart contract allows the Regulatory Authority to update the candidates profile
5. newVote
   1. This smart contract grabs the voter’s Emirates ID, the chosen candidate and registers the new vote. It also creates a Vote asset that is linked with the voter’s Emirates ID, sets the voter’s ‘hasVoted’ to true and increments the candidates ‘numVotes’ by 1.

All these smart contract transactions are updated in the ledger and can not be reversed.

# Testing

## Functional Testing Goals

The goal of testing the different functionalities of our system TasweetX, is to ensure that once the system is implemented it will have no errors. This document is the result of the testing phase of our functional requirements wherein all the Voter and Regulatory Authority requirements from the Websites were taken into consideration and tested using different scenarios that the user might face while using the system.

## Functional Test Plan Scope

### In Scope

All Voter and Regulatory Authority requirements are being tested in this phase.

### Out of Scope

Non-functional requirements like security and performance are not being tested in this phase, these will be tested in the next phase.

## Functional Test Plan Assumptions and Constraints

### Assumptions

For all test cases, we assume that:

* The user has a working internet connection throughout
* Has access to the Website (Regulatory Authority and Voters)
* The Regulatory Authority has the Username and Password that was given by the Admin
* The Voters have valid Emirates IDs, and are eligible to register/ vote

### Constraints

At any point, if the system faces the following issues, the test cases may not be successful

* Lose internet connection
* Not all scenarios can be tested
* All test cases can only be tested by one person integrating the system due to remote working

## Functional Test Team Roles and Responsibilities

|  |  |  |
| --- | --- | --- |
| **Name** | **Role** | **Responsibility** |
| Ayesha Riyaz | Website Testing | Find possible test cases to test system to the max and note down expected results |
| Saher Hasan | Website Testing | Find possible test cases to test system to the max and note down expected results |
| Mahmoud AlSofyani | Blockchain Integration into Website | Find possible test cases to test system to the max and note down expected results |
| Farhana Islam | Website Testing | Find possible test cases to test system to the max and note down expected results |

## Functional Requirement Test Cases

## 1.1 User (Voter) requirements

These requirements are for the users (voters) of the website and the following section will list the test cases for each requirement

### 1.1.1 Login

|  |  |
| --- | --- |
| Tested requirement | *Logging in – 1.1.1.1* |
| Quick description | This test case checks the login functionality if the registered voter tries logging in using valid username and password |
| Pre-requisite | Valid username and password that is registered in the system  Connected to the internet |
| Input | Valid username and password entered |
| Expected output | Successful Login |
| Observed output | Login was successful |
| Verdict | No changes required |
| Comments | User has to login with full email address |

|  |  |
| --- | --- |
| Tested requirement | *Logging in – 1.1.1.2* |
| Quick description | This test case checks the login functionality if the registered voter tries logging in using in invalid username and password |
| Pre-requisite | Username and password that is not registered in the system  Connected to the internet |
| Input | Invalid username and password entered |
| Expected output | An error message saying “The account or password entered is incorrect” is displayed to the voter |
| Observed output | Error Message is displayed, and the page does not advance |
| Verdict | No changes required |
| Comments | Might add a message in a future version that tells the user to double check their email if they have made a typo like .con or forgot the @ symbol. |

|  |  |
| --- | --- |
| Tested requirement | *Logging in – 1.1.1.3* |
| Quick description | This test case checks the login functionality when the voter tries to log in leaving username and password empty |
| Pre-requisite | Connected to the internet |
| Input | No input entered |
| Expected output | A message saying “Please enter username and password to login” is displayed to the voter |
| Observed output | A message saying “Please enter username and password to login” is displayed to the voter |
| Verdict | No changes required |
| Comments | n/a |

|  |  |
| --- | --- |
| Tested requirement | *Logging in – 1.1.1.4* |
| Quick description | This test case checks if the voter tries to login again using correct credentials, after logging out |
| Pre-requisite | Voter must be logged out of the system.  Connected to the internet |
| Input | Valid username and password entered  Click login button |
| Expected output | Successful Login |
| Observed output | Login successful |
| Verdict | No changes required |
| Comments | n/a |

### 1.1.2 Logout

|  |  |
| --- | --- |
| Tested requirement | *Logging out – 1.1.2.1* |
| Quick description | This test case checks the logout functionality when the voter tries to logout from his account |
| Pre-requisite | Voter must be logged in the system.  Connected to the internet |
| Input | Selecting the log out button |
| Expected output | The screen is redirected to the Login page |
| Observed output | The screen is redirected to the Login Page |
| Verdict | No changes required |
| Comments | n/a |

|  |  |
| --- | --- |
| Tested requirement | *Logging out – 1.1.2.2* |
| Quick description | This test case checks the consequence when the voter selects the log out button, the voter should not be able to access any of the account or voter dashboard details |
| Pre-requisite | Voter must be logged in the system.  Connected to the internet |
| Input | Selecting the log out button |
| Expected output | A message is displayed to the voter saying “You have been logged out.”  the screen is redirected to the login page |
| Observed output | The screen is redirected to the login page which shows no user specific functions |
| Verdict | No changes required |
| Comments | n/a |

### 1.1.3 Create Account

|  |  |
| --- | --- |
| Tested requirement | *Create Account – 1.1.3.1* |
| Quick description | This test case checks the register functionality when the user tries to create an account in the app, with a valid Emirates ID in order to vote. |
| Pre-requisite | The user is the voter for the election period.  The user must enter a valid Emirates ID to be an eligible voter  The user should accept the terms & conditions  Connected to the internet |
| Input | Selecting the register button |
| Expected output | Successful account creation and registered/eligible voter |
| Observed output | Profile created with no issues |
| Verdict | There is no API that verifies if the emirates ID is valid or not. |
| Comments | We automatically assume the emirates ID is valid if it conforms to our regex pattern |

|  |  |
| --- | --- |
| Tested requirement | *Create Account – 1.1.3.2* |
| Quick description | This test case checks the register functionality when the user tries to create an account in the app, with an invalid Emirates ID |
| Pre-requisite | The user is the voter for the election period.  The user enters invalid Emirates ID  The user should accept the terms & conditions  Connected to the internet |
| Input | Selecting the register button |
| Expected output | A message saying “Please enter a valid Emirates ID in order to be an eligible voter” is displayed to the voter |
| Observed output | Entered emirates ID number is rejected |
| Verdict | The only way we verify an emirates ID is through the regex pattern |
| Comments | n/a |

### 1.1.4 Manage Profile Page

1.1.4.1 Modify Profile

|  |  |
| --- | --- |
| Tested requirement | *Modify Profile – 1.1.4.1.1* |
| Quick description | This test case checks the manage profile functionality when the voter wants to manage their profile account. |
| Pre-requisite | The voter must be logged in  Connected to the internet |
| Input | Selecting the manage profile option in the side menu bar |
| Expected output | the screen is redirected to the Manage Profile Page |
| Observed output | Screen redirects successfully |
| Verdict | No changes required |
| Comments | n/a |

|  |  |
| --- | --- |
| Tested requirement | *Modify Profile – 1.1.4.1.2* |
| Quick description | This test case checks if the voter can modify their profile details when he clicks the save changes button. |
| Pre-requisite | Voter must be logged in  Voter is on the Manage Profile Page  Connected to the internet |
| Input | Selecting the save changes button |
| Expected output | Changes saved successfully |
| Observed output | Profile changes are saved properly |
| Verdict | No changes required |
| Comments | Future version may feature a notification that changes have been saved |

### 1.1.5 Candidate Dashboard Page

1.1.5.1 View Candidate List

|  |  |
| --- | --- |
| Tested requirement | *View Candidate list – 1.1.5.1.1* |
| Quick description | This test case checks the Candidate Dashboard functionality when the voter wants to view candidate list. |
| Pre-requisite | Voter must be logged in  Connected to the internet |
| Input | Selecting the Candidate Dashboard button in the side menu bar |
| Expected output | The screen is redirected to the Candidate Dashboard page  The candidate list is displayed. |
| Observed output | Candidate list is displayed with no issues |
| Verdict | No changes required |
| Comments | n/a |

### 1.1.6 Selected Candidate Page

1.1.6.1 View Candidate Description

|  |  |
| --- | --- |
| Tested requirement | *View Candidate Description – 1.1.6.1.2* |
| Quick description | This test case checks if the voter can view selected candidate descriptions from the candidate list. |
| Pre-requisite | Voter must be logged in  Voter is on the Candidate Dashboard Page  Connected to the Internet |
| Input | Select the candidate to be voted for |
| Expected output | Candidate description is displayed to the voter |
| Observed output | Candidate description and photo is displayed with no issues |
| Verdict | No changes required |
| Comments | n/a |

|  |  |
| --- | --- |
| Tested requirement | *View Candidate Description – 1.1.6.1.3* |
| Quick description | This test case checks if the voter can view another selected candidate description once he has already selected a candidate previously to view from the candidate list. |
| Pre-requisite | Voter must be logged in  Voter is on the Candidate Dashboard Page  Connected to the Internet  Selection of one candidate already done |
| Input | Close Selected Candidate Page and select another candidate to be voted for |
| Expected output | New candidate description is displayed to the voter |
| Observed output | New candidate description and photo is displayed properly |
| Verdict | No changes required |
| Comments | n/a |

1.1.6.2 Cast Vote

|  |  |
| --- | --- |
| Tested requirement | *Cast Vote– 1.1.6.2.1* |
| Quick description | This test case checks if the eligible voter can cast a vote on the selected candidate from the candidate list. |
| Pre-requisite | Voter must be logged in  Voter has selected a candidate  Connected to the internet  Smart contract approves vote |
| Input | Click the cast vote button |
| Expected output | Vote casted and added to the ledger |
| Observed output | Vote was casted to ledger |
| Verdict | Vote has a transaction ID and timestamp |
| Comments | n/a |

|  |  |
| --- | --- |
| Tested requirement | *Cast Vote– 1.1.6.2.2* |
| Quick description | This test case checks the consequence when the voter tries to vote for the second time. |
| Pre-requisite | Voter must be logged in  Connected to the internet  Voter has already cast a vote, and the vote has been saved as a transaction. |
| Input | Select the candidate to vote for |
| Expected output | Cast vote button disabled. |
| Observed output | Can not cast another vote |
| Verdict | Might add a notification that is User friendly that shows a message |
| Comments | n/a |

### 1.1.7 Voter Dashboard Page

1.1.7.1 View Vote History

|  |  |
| --- | --- |
| Tested requirement | *View Vote History – 1.1.7.1* |
| Quick description | This test case checks if the voter can view vote history when he wants to view vote statistics. |
| Pre-requisite | Voter must be logged in  Voter has cast a vote  Connected to the internet |
| Input | Click on the View Vote history option in the side menu |
| Expected output | The screen is redirected to the Voter Dashboard Page  Vote history and voting statistics displayed |
| Observed output | Voter vote history displays with no issues |
| Verdict | No changes required |
| Comments | n/a |

## 1.2 Regulatory Authority requirements

These requirements are for the regulatory authority of the website and the following section will list the test cases for each requirement

### 1.2.1 Login

|  |  |
| --- | --- |
| **Tested requirement** | ***Logging in – 1.2.1.1*** |
| **Quick description** | This test case checks the login functionality if the registered regulatory authority logs in using valid username and password |
| **Pre-requisite** | Valid username and password that is registered in the system  Connected to the internet |
| **Input** | Valid username and password entered |
| **Expected output** | Successful Login |
| **Observed output** | Login was successful |
| **Verdict** | No changes required |
| **Comments** | n/a |

|  |  |
| --- | --- |
| **Tested requirement** | ***Logging in – 1.2.1.2*** |
| **Quick description** | This test case checks the login functionality if the regulatory authority tries logging in using in invalid username and password |
| **Pre-requisite** | Username and password that is not registered in the system  Connected to the internet |
| **Input** | Invalid username and password entered |
| **Expected output** | An error message saying “The account username or password entered is incorrect” displayed to the regulatory authority  Regulatory authority access not granted |
| **Observed output** | Username or password was incorrect |
| **Verdict** | No changes required |
| **Comments** | n/a |

|  |  |
| --- | --- |
| **Tested requirement** | ***Logging in – 1.2.1.3*** |
| **Quick description** | This test case checks the forgot password functionality if the regulatory authority while logging in has forgotten password |
| **Pre-requisite** | Connected to the internet  Valid Username entered |
| **Input** | Forgot Password clicked |
| **Expected output** | Redirected to Forgot Password Page to set a new Password |
| **Observed output** | Redirected to Forgot Password Page to set a new Password |
| **Verdict** | No changes required |
| **Comments** | n/a |

|  |  |
| --- | --- |
| **Tested requirement** | ***Logging in – 1.2.1.4*** |
| **Quick description** | This test case checks the set new password functionality if the registered regulatory authority sets new password |
| **Pre-requisite** | Valid username and password that is registered in the system  Connected to the internet  Forgot password option selected |
| **Input** | New Password |
| **Expected output** | Password updated |
| **Observed output** | Updates password |
| **Verdict** | No changes required |
| **Comments** | n/a |

1.2.2 Logout

|  |  |
| --- | --- |
| **Tested requirement** | ***Logging out – 1.2.2.1*** |
| **Quick description** | This test case checks the logout functionality when the regulatory authority logs out from their account |
| **Pre-requisite** | Regulatory Authority must be logged in the system.  Connected to the internet |
| **Input** | Selecting the log out button |
| **Expected output** | Successful logout from their account and redirected to login page |
| **Observed output** | Successful log out |
| **Verdict** | No changes required |
| **Comments** | n/a |

### 1.2.3 Start Election

|  |  |
| --- | --- |
| **Tested requirement** | ***Start Elections – 1.2.3.1*** |
| **Quick description** | This test case checks the functionality of regulatory authorities’ ability to “start” that indicates the beginning of elections for voters to cast vote. |
| **Pre-requisite** | Regulatory Authority must be logged in the system.  Actual elections to be held loaded with candidate’s database  Connected to the internet |
| **Input** | Selecting the start elections button |
| **Expected output** | Successful beginning of elections. |
| **Observed output** | Starting elections successful |
| **Verdict** | No changes required |
| **Comments** | n/a |

### 1.2.4 End Elections

|  |  |
| --- | --- |
| **Tested requirement** | ***End Elections – 1.2.4.1*** |
| **Quick description** | This test case checks the functionality of regulatory authorities’ ability to “end” elections that indicates the closure of elections meaning no more votes can be casted any more |
| **Pre-requisite** | Regulatory Authority must be logged in the system.  Actual elections to be held loaded with candidate’s database  Connected to the internet  Voters casting their votes |
| **Input** | Selecting the end elections button |
| **Expected output** | Successful ending of elections. |
| **Observed output** | Ending elections successful |
| **Verdict** | No changes required |
| **Comments** | n/a |

### 1.2.5 Add Candidate

|  |  |
| --- | --- |
| **Tested requirement** | ***Add candidate – 1.2.5.1*** |
| **Quick description** | This test case checks the add candidate functionality that lets regulatory authority to add new candidates standing for elections |
| **Pre-requisite** | Regulatory Authority must be logged in the system.  Connected to the internet  Information of new candidate  Information of the candidate to be verified |
| **Input** | Selecting the Add Candidate button  Enter details in the Add Candidate form  Name  Email  Emirates Id  Description |
| **Expected output** | Successful input of details of the new candidate  Candidate profile displayed on the Candidates page |
| **Observed output** | Adding candidate processed properly |
| **Verdict** | No changes required |
| **Comments** | n/a |

### 1.2.6 Delete Candidate

|  |  |
| --- | --- |
| **Tested requirement** | ***Delete Candidate 1.2.6.1*** |
| **Quick description** | This test case checks the delete candidate functionality that lets regulatory authority to remove candidates profile from the database |
| **Pre-requisite** | Regulatory Authority must be logged in the system.  Connected to the internet  Existing Candidate Profile |
| **Input** | Selecting the delete button |
| **Expected output** | Successful removal of candidate profile |
| **Observed output** | Deleting candidate from database processed properly |
| **Verdict** | No changes required |
| **Comments** | n/a |

### 1.2.7 Edit Candidate Profile

|  |  |
| --- | --- |
| **Tested requirement** | ***Edit Candidate Profile – 1.2.7.1*** |
| **Quick description** | This test case checks the edit functionality for the regulatory authority to modify information of existing candidates. |
| **Pre-requisite** | Regulatory Authority must be logged in the system.  Connected to the internet  Existing Candidate Profile |
| **Input** | Change candidate information  Select the Save Changes button |
| **Expected output** | Successful change of the Candidate Profile |
| **Observed output** | Editing candidate profile by RA done properly |
| **Verdict** | No changes required |
| **Comments** | n/a |

|  |  |
| --- | --- |
| **Tested requirement** | ***Edit Candidate Profile – 1.2.7.2*** |
| **Quick description** | This test case checks the cancel functionality for the regulatory authority in case there is no edit made . |
| **Pre-requisite** | Regulatory Authority must be logged in the system.  Connected to the internet  Existing Candidate Profile |
| **Input** | No input of candidate information  Select the Save Changes button |
| **Expected output** | Candidate Profile unchanged |
| **Observed output** | No changes made to the profile |
| **Verdict** | No changes made |
| **Comments** | n/a |

### 1.2.8 Tally Votes

|  |  |
| --- | --- |
| **Tested requirement** | ***Tally Votes – 1.2.8.1*** |
| **Quick description** | This test case checks the tally functionality when the regulatory authority tallies the number of votes casted for an election |
| **Pre-requisite** | Regulatory Authority must be logged in the system.  Connected to the internet  Elections started  Election running  Voters casting their votes |
| **Input** | Selecting the tally button |
| **Expected output** | Successful counting of votes |
| **Observed output** | Correct number of votes displayed |
| **Verdict** | No changes required |
| **Comments** | n/a |

### 1.2.9 Announce Results

|  |  |
| --- | --- |
| **Tested requirement** | ***Announce Results – 1.2.9.1*** |
| **Quick description** | This test case checks the announce results functionality when the regulatory authority wants to display the results of the elections |
| **Pre-requisite** | Regulatory Authority must be logged in the system.  Connected to the internet  Elections started  Election running  Votes casting their votes  Elections end  Tallying of votes |
| **Input** | Selecting the announce results button |
| **Expected output** | Successful display of the winner of the election |
| **Observed output** | Displays winning candidate |
| **Verdict** | Might add email notification to send an email to all voters on who the winner is. But feature is working fine as is |
| **Comments** | n/a |

## Functional Testing Results

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Name** | **Pass/Fail** | **Date Tested** |
| T01 | Logging in 1.1.1.1 | Pass | 05/01/2020 |
| T02 | Logging in 1.1.1.2 | Pass | 05/01/2020 |
| T03 | Logging in 1.1.1.3 | Pass | 05/01/2020 |
| T04 | Logging in 1.1.1.4 | Pass | 05/01/2020 |
| T05 | Logging out 1.1.2.1 | Pass | 05/01/2020 |
| T06 | Logging out 1.1.2.2 | Pass | 05/01/2020 |
| T07 | Create Account 1.1.3.1 | Pass | 05/01/2020 |
| T08 | Create Account 1.1.3.2 | Pass | 05/01/2020 |
| T09 | Modify Profile 1.1.4.1.1 | Pass | 05/01/2020 |
| T10 | Modify Profile 1.1.4.1.2 | Pass | 05/01/2020 |
| T11 | View Candidate List 1.1.5.1.1 | Pass | 05/04/2020 |
| T12 | View Candidate Description 1.1.6.1.2 | Pass | 05/04/2020 |
| T13 | View Candidate Description 1.1.6.1.3 | Pass | 05/04/2020 |
| T14 | Cast Vote 1.1.6.2.1 | Pass | 05/04/2020 |
| T15 | Cast Vote 1.1.6.2.2 | Pass | 05/04/2020 |
| T16 | View Vote History 1.1.7.1 | Pass | 05/04/2020 |
| T17 | Logging in 1.2.1.1 | Pass | 05/01/2020 |
| T18 | Logging in 1.2.1.2 | Pass | 05/01/2020 |
| T19 | Logging in 1.2.1.3 | Pass | 05/01/2020 |
| T20 | Logging in 1.2.1.4 | Pass | 05/01/2020 |
| T21 | Logging out 1.2.2.1 | Pass | 05/01/2020 |
| T22 | Start Elections 1.2.3.1 | Pass | 05/04/2020 |
| T23 | End Elections 1.2.4.1 | Pass | 05/04/2020 |
| T24 | Add Candidate 1.2.5.1 | Pass | 05/04/2020 |
| T25 | Delete Candidate 1.2.6.1 | Pass | 05/04/2020 |
| T26 | Edit Candidate 1.2.7.1 | Pass | 05/04/2020 |
| T27 | Edit Candidate 1.2.7.2 | Pass | 05/04/2020 |
| T28 | Tally Votes 1.2.8.1 | Pass | 05/04/2020 |
| T29 | Announce Results 1.2.9.1 | Pass | 05/04/2020 |

## Traceability Matrix

### Traceability Matrix for Voters’ Requirements

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Requirement ID** | **T01** | **T02** | **T03** | **T04** | **T05** | **T06** | **T07** | **T08** | **T09** | **T10** | **T11** | **T12** | **T13** | **T14** | **T15** | **T16** |
| R001 | Checkmark | Checkmark | Checkmark | Checkmark | Checkmark | Checkmark |  |  |  |  |  |  |  |  |  |  |
| R002 |  |  |  |  | Checkmark | Checkmark |  |  |  |  |  |  |  |  |  |  |
| R003 |  |  |  |  |  |  | Checkmark | Checkmark |  |  |  |  |  |  |  |  |
| R004 |  |  |  |  |  |  |  |  | Checkmark | Checkmark |  |  |  |  |  |  |
| R005 |  |  |  |  |  |  |  |  |  |  | Checkmark | Checkmark | Checkmark | Checkmark | Checkmark | Checkmark |
| R006 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | Checkmark |

\*\*refer to the appendix below for Requirement ID and Testcase ID reference

### Traceability Matrix for RA’s Requirements

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Requirement ID** | **T17** | **T18** | **T19** | **T20** | **T21** | **T22** | **T23** | **T24** | **T25** | **T26** | **T27** | **T28** | **T29** |
| R007 | Checkmark | Checkmark | Checkmark | Checkmark | Checkmark |  |  |  |  |  |  |  |  |
| R08 |  |  |  |  |  |  |  | Checkmark |  |  |  |  |  |
| R009 |  |  |  |  |  |  |  |  | Checkmark |  |  |  |  |
| R010 |  |  |  |  |  |  |  |  |  | Checkmark | Checkmark |  |  |
| R011 |  |  |  |  |  | Checkmark |  |  |  |  |  |  |  |
| R012 |  |  |  |  |  |  | Checkmark |  |  |  |  |  |  |
| R013 |  |  |  |  |  |  |  |  |  |  |  | Checkmark |  |
| R014 |  |  |  |  |  |  |  |  |  |  |  |  |  |
| R015 |  |  |  |  |  |  |  |  |  |  |  |  | Checkmark |

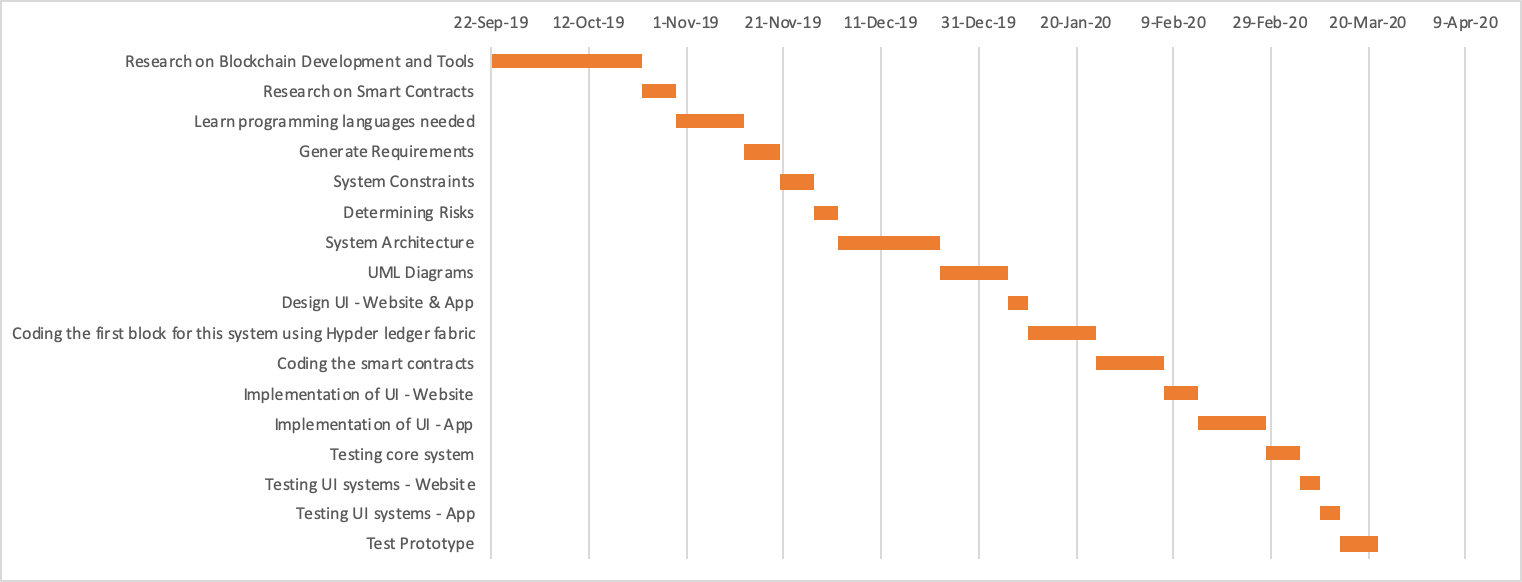
\*\*refer to the appendix below for Requirement ID and Testcase ID reference

# Maintenance

# Appendices

# Feasibility:

### Gantt Chart



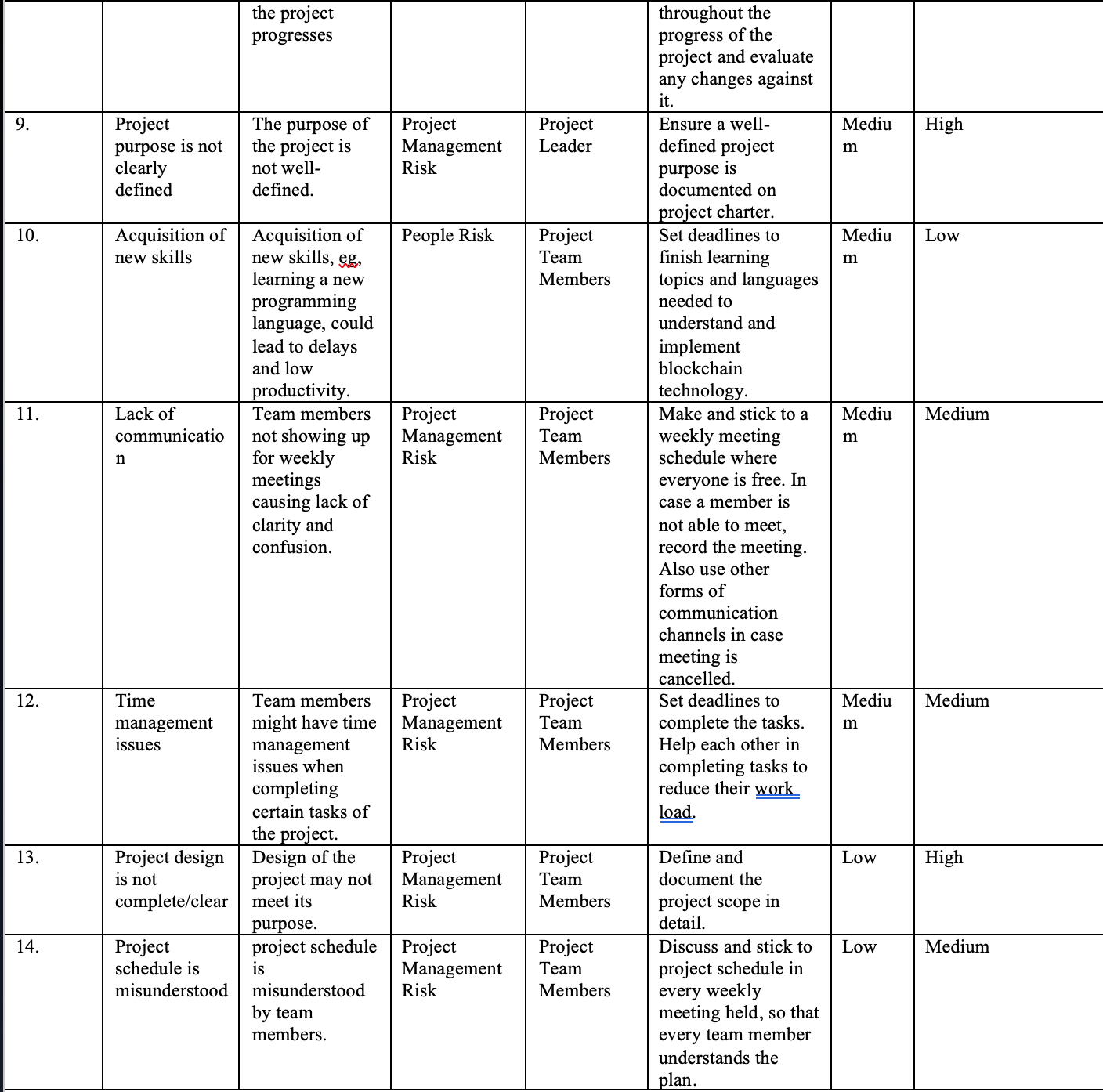
# Requirements: Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Name** | **Description** |
| 1.0 | 29/10/2019 | Preliminary Requirements | This version lists out the preliminary requirements for TasweetX |

### Requirements Traceability Matrix

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Requirement No | Name | Category | Source | Status |
| R01 | Laptop Memory | Hardware | Project Charter and Laptop specifications | Complete. Laptops have 8GB memory meeting requirements. |
| R02 | Login to the system | Software | Specifications of the software | In Progress. |
| R03 | Cast Vote | Software | Specifications of the software | In Progress. |
| R04 | Check their vote | Software | Specifications of the software | In Progress. |
| R05 | Log out | Software | Specifications of the software | In Progress. |
| R06 | Count votes | Hardware and Software | Project charter and software specifications for computational power | On Hold |

### Qualitative Risk Analysis



# Design:

### Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Primary Authors | Description | Date Completed |
| v1.0 | TasweetX | Initial design specifications for TasweetX Blockchain voting system | 08/12/2019 |
| v2.0 | TasweetX | Modified and Final design specifications for TasweetX Blockchain voting system | 04/05/2020 |

# Testing:

### Requirement list

|  |  |
| --- | --- |
| **Requirement ID** | **Requirement Name** |
| R001 | Login to Voter account |
| R002 | Logout of Voter account |
| R003 | Voter creates an account to vote |
| R004 | Voter modifies their profile |
| R005 | Voter casts vote |
| R006 | Voter views vote history |
| R007 | Login to Regulatory Authority account |
| R008 | Regulatory Authority adds candidate profile |
| R009 | Regulatory Authority deletes candidate profile |
| R010 | Regulatory Authority edits candidate profile |
| R011 | Regulatory Authority starts election |
| R012 | Regulatory Authority ends election |
| R013 | Regulatory Authority tallies votes |
| R014 | Regulatory Authority observes votes |
| R015 | Regulatory Authority announces results |

### Testcase List

|  |  |
| --- | --- |
| **Test ID** | **Test Name** |
| T01 | Logging in 1.1.1.1 |
| T02 | Logging in 1.1.1.2 |
| T03 | Logging in 1.1.1.3 |
| T04 | Logging in 1.1.1.4 |
| T05 | Logging out 1.1.2.1 |
| T06 | Logging out 1.1.2.2 |
| T07 | Create Account 1.1.3.1 |
| T08 | Create Account 1.1.3.2 |
| T09 | Modify Profile 1.1.4.1.1 |
| T10 | Modify Profile 1.1.4.1.2 |
| T11 | View Candidate List 1.1.5.1.1 |
| T12 | View Candidate Description 1.1.6.1.2 |
| T13 | View Candidate Description 1.1.6.1.3 |
| T14 | Cast Vote 1.1.6.2.1 |
| T15 | Cast Vote 1.1.6.2.2 |
| T16 | View Vote History 1.1.7.1 |
| T17 | Logging in 1.2.1.1 |
| T18 | Logging in 1.2.1.2 |
| T19 | Logging in 1.2.1.3 |
| T20 | Logging in 1.2.1.4 |
| T21 | Logging out 1.2.2.1 |
| T22 | Start Elections 1.2.3.1 |
| T23 | End Elections 1.2.4.1 |
| T24 | Add Candidate 1.2.5.1 |
| T25 | Delete Candidate 1.2.6.1 |
| T26 | Edit Candidate 1.2.7.1 |
| T27 | Edit Candidate 1.2.7.2 |
| T28 | Tally Votes 1.2.8.1 |
| T29 | Announce Results 1.2.9.1 |

## Skill Tracker





## Meeting Minutes

*Meeting Minutes*

Week: #2\_\_\_\_\_\_\_\_\_ Date: 22/09/19 Time: \_\_\_12:30-13:00 PM

Present

1. Saher Hasan
2. Ayesha Riyaz
3. Farhana Islam
4. Mahmoud AlSofyani

Absent: none

|  |  |  |  |
| --- | --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Resolution | Date of Resolution |
| 1 | Discuss additional ideas | See if the team can come up with more ideas | Till end of week2 |
| 2 | Contract Signing | To submit to Dr Zeenath | 22/09/19 |
| 3 | Domain for blockchain |  | Week2 |
| 4 | In depth learning- blockchain | Discuss and individually understand how blockchain works | Throughout the semester |
| 5 | Prepare for meeting with Dr Zeenath |  | 22/09/19 |

Next meeting date: 23/09/19

Proposed agenda: Further discussion on Idea1 and potential ideas for the other two options

|  |  |  |
| --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Raised by |
| 1 | Come up with two more ideas | team |
| 2 | Decide potential mentor | team |
| 3 | Choose a team name and possible project names | Ayesha, Mahmoud |

*Meeting Minutes*

Week: #\_\_\_\_\_2\_\_\_\_ Date: 23/09/19 Time: 12:30-13:30PM

Present

1. Saher Hasan
2. Ayesha Riyaz
3. Farhana Islam
4. Mahmoud AlSofyani

Absent: none

|  |  |  |  |
| --- | --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Resolution | Date of Resolution |
| 1 | Pick a team name | Name team | By end of week 2 |
| 2 | Proposal Forms | Be clear about what the proposal forms are and how to go about filling them | By end of week 2 |
| 3 | Proposal Form1- picked idea, researched | Know more about idea 1 before writing the proposal | By end of week 2 |
| 4 | Discuss possible mentors: who, why | To be able to pick the right mentor | By end of week 3 |

Next meeting date: 25/09/19

Proposed agenda: Discuss idea 2 and 3 and finalize

|  |  |  |
| --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Raised by |
| 1 | questions | team |
| 2 | research | team |

*Meeting Minutes*

Week: #\_\_\_2\_\_\_\_\_\_ Date: 25/09/19 Time: 12:30-14:30PM

Present

1. Saher Hasan
2. Ayesha Riyaz
3. Farhana Islam
4. Mahmoud AlSofyani

Absent: none

|  |  |  |  |
| --- | --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Resolution | Date of Resolution |
| 1 | Questions regarding weekly submissions | To be able to submit weekly blogs efficiently | End of week 2 |
| 2 | Questions regarding proposal forms, feasibility study and ranking | Planning the following weeks in advance | 25/09/19 |
| 3 | Research on visual cryptography | For possible implementation in project- checking scope | Following weeks |
| 4 | Discuss idea 2 | Finalize idea 2 | 25/09/19 |

Next meeting date: tbd

Proposed agenda: Discuss and start working on proposals for all three ideas

|  |  |  |
| --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Raised by |
| 1 | Proposal 1, 2 and 3 | team |
| 2 | Additional research on domains for the three ideas | team |
| 3 | Presentation of proposals: how to go about it | team |

*Meeting Minutes*

Week: #\_\_3\_\_\_\_\_\_\_ Date: 02/10/19 Time: 12:30-14:30PM

Present

1. Saher Hasan
2. Mahmoud AlSofyani
3. Farhana Islam
4. Ayesha Riyaz

Absent - none

|  |  |  |  |
| --- | --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Resolution | Date of Resolution |
| 1 | Finalize, format and print proposals | Finalize proposals | 02/10/19 |

Next meeting date: tbd

Proposed agenda:

|  |  |  |
| --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Raised by |
| 1 | Set meeting dates for coming week | Farhana |

*Meeting Minutes*

Week: #\_\_\_\_3\_\_\_\_\_ Date: 03/10/19 Time: 12:30-13:30PM

Present

1. Saher Hasan

2. Farhana Islam

3. Ayesha Riyaz

Absent

1. Mahmoud AlSofyani

|  |  |  |  |
| --- | --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Resolution | Date of Resolution |
| 1 | Decide dates for group studying on how blockchain works | To learn more about the project topic | Following few weeks |

Next meeting date: tbd

Proposed agenda:

|  |  |  |
| --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Raised by |
| 1 | Start preparing for proposal presentations | team |
| 2 | Start working on feasibility study outline | Saher, Farhana |

*Meeting Minutes*

Week: #\_\_\_\_5\_\_\_\_\_ Date: 10/10/19 Time: 12:30-14:30PM

Present

1. Saher Hasan
2. Mahmoud AlSofyani
3. Farhana Islam
4. Ayesha Riyaz

Absent - none

|  |  |  |  |
| --- | --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Resolution | Date of Resolution |
| 1 | Start working on feasibility report- divided parts for each | Start feasibility report | tbd |

Next meeting date: tbd

Proposed agenda:

|  |  |  |
| --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Raised by |
| 1 | Discuss updates on Feasibility report- everyone’s parts | all |

*Meeting Minutes*

Week: #\_\_\_\_\_6\_\_\_\_ Date: 18/10/19 Time: 12:30-13:30PM

Present

1. Saher Hasan

2. Farhana Islam

3. Ayesha Riyaz

4. Mahmoud Al Sofyani

|  |  |  |  |
| --- | --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Resolution | Date of Resolution |
| 1 | Discuss updates on feasibility report and helped each other on parts that were difficult | Finalize Feasibility document | Next week |

Next meeting date: tbd

Proposed agenda:

|  |  |  |
| --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Raised by |
| 1 | Prepare for Requirements document | team |
| 2 | Start preparing for presentation | Saher, Farhana |

*Meeting Minutes*

Week: #\_\_\_\_\_7\_\_\_\_ Date: 25/10/19 Time: 12:30-13:30PM

Present

1. Saher Hasan

2. Farhana Islam

3. Ayesha Riyaz

4. Mahmoud Al Sofyani

|  |  |  |  |
| --- | --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Resolution | Date of Resolution |
| 1 | Start working on requirements document- divide parts | Finish requirements document | Next week |

Next meeting date: tbd

Proposed agenda:

|  |  |  |
| --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Raised by |
| 1 | Finish Requirements document | team |

*Meeting Minutes*

Week: #\_\_\_\_\_8\_\_\_\_ Date: 04/11/19 Time: 12:30-13:30PM

Present

1. Saher Hasan

2. Farhana Islam

3. Ayesha Riyaz

4. Mahmoud Al Sofyani

|  |  |  |  |
| --- | --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Resolution | Date of Resolution |
| 1 | Discuss the prototype of the application and website | Come up with a common idea of interfaces | Following weeks |

Next meeting date: tbd

Proposed agenda:

|  |  |  |
| --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Raised by |
| 1 | Decide which software to use for UI design and development | team |

*Meeting Minutes*

Week: #\_\_\_\_\_10\_\_\_\_ Date: 18/11/19 Time: 12:30-13:30PM

Present

1. Saher Hasan

2. Farhana Islam

3. Ayesha Riyaz

4. Mahmoud Al Sofyani

|  |  |  |  |
| --- | --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Resolution | Date of Resolution |
| 1 | Blockchain development discussion- how it can be implemented | Check for feasibility again, be sure no changes are to be made | immediate |
| 2 | Discussion on blockchain mechanisms from an implementation point of view | Clarity on blockchain | immediate |
| 3 | Design document preparation | To start design document | Following weeks |

Next meeting date: tbd

Proposed agenda:

|  |  |  |
| --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Raised by |
| 1 | Prepare for Design document and presentation | team |

*Meeting Minutes*

Week: #break Date: 09/01/2020 Time: 12:30-13:30PM

Present

1. Saher Hasan

2. Farhana Islam

3. Ayesha Riyaz

4. Mahmoud Al Sofyani

|  |  |  |  |
| --- | --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Resolution | Date of Resolution |
| 1 | Updates on Proof of concept presentation | Be prepared for proof of concept | 1st week of second sem |
| 2 | Divide work for each role | POC presentation | immediate |

Next meeting date: tbd

Proposed agenda:

|  |  |  |
| --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Raised by |
| 1 | POC finalizing | team |

*Meeting Minutes*

Week: #\_\_\_1

Present

1. Saher Hasan

2. Farhana Islam

3. Ayesha Riyaz

4. Mahmoud Al Sofyani

|  |  |  |  |
| --- | --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Resolution | Date of Resolution |
| 1 | Finalize POC presentation updates | Present the POC | immediate |

Next meeting date: tbd

Proposed agenda:

|  |  |  |
| --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Raised by |
| 1 | Design Document Updated version | team |

*Meeting Minutes*

Week: #\_\_\_\_\_5\_\_\_\_

Present

1. Saher Hasan

2. Farhana Islam

3. Ayesha Riyaz

4. Mahmoud Al Sofyani

|  |  |  |  |
| --- | --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Resolution | Date of Resolution |
| 1 | Implementation Updates – Website and Blockchain | Try to finish implementation | Following weeks |
| 2 | Dropped the idea of Application due to difficulty in implementation | - | - |

Next meeting date: tbd

Proposed agenda:

|  |  |  |
| --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Raised by |
| 1 | Finalize implementation | team |

*Meeting Minutes*

Week: #\_\_\_\_\_10\_\_\_\_

Present

1. Saher Hasan

2. Farhana Islam

3. Ayesha Riyaz

4. Mahmoud Al Sofyani

|  |  |  |  |
| --- | --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Resolution | Date of Resolution |
| 1 | Updates on team member’s progress, discussed testing possibilities | To see if anything is lacking | Following weeks |

Next meeting date: tbd

Proposed agenda:

|  |  |  |
| --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Raised by |
| 1 | Finalize website pages and blockchain implementation, prepare for testing | team |

*Meeting Minutes*

Week: #\_\_\_\_\_11\_\_\_

Present

1. Saher Hasan

2. Farhana Islam

3. Ayesha Riyaz

4. Mahmoud Al Sofyani

|  |  |  |  |
| --- | --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Resolution | Date of Resolution |
| 1 | Updates on team member’s progress | To see if anything is lacking | Following weeks |
| 2 | Submit testing documents 1&2 | Submission | Same week |

Next meeting date: tbd

Proposed agenda:

|  |  |  |
| --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Raised by |
| 1 | Finalize website pages and blockchain implementation | team |

*Meeting Minutes*

Week: #\_\_\_\_\_12\_\_\_\_

Present

1. Saher Hasan

2. Farhana Islam

3. Ayesha Riyaz

4. Mahmoud Al Sofyani

|  |  |  |  |
| --- | --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Resolution | Date of Resolution |
| 1 | Updates on team member’s progress- compile final document | Update | Next week |

Next meeting date: tbd

Proposed agenda:

|  |  |  |
| --- | --- | --- |
| SR | Agenda and Issues to be Discussed | Raised by |
| 1 | Finalize website pages and blockchain implementation-updates | team |

# References

1. Blockchain jobs and salaries - 2018 report. (n.d.). Retrieved October 15, 2019, from <https://hackernoon.com/blockchain-jobs-and-salaries-2018-report-45d3e7741c19>.
2. App Developer Salary Guide (2018). (n.d.). Retrieved October 15, 2019, from <https://www.businessofapps.com/app-developers/research/ios-android-developer-salary/>.
3. Stakeholdermap.com. (n.d.). *20 Common Project Risks - example Risk Register*. [online] Available at: https://www.stakeholdermap.com/risk/register-common-project-risks.html [Accessed 15 Oct. 2019].