****

**Pakistan Air Force**

**Karachi Institute of Economics & Technology**

**Blockchain Election Voting System(Devote)**

**Submitted by:**

**Saad Abbas**

62908

**Syed Ebadullah**

8979

**Advisor:**

**Sir Muneeb**

College of Computing & Information Sciences

A final year project report presented to the College of Computing & Information Sciences,  
Pakistan Air Force – Karachi Institute of Economics & Technology in partial fulfillment of the requirements of the degree of Bachelor of Science

**2020-2021**

Contents

[Abstract iii](#_Toc76737085)

[Acknowledgments vi](#_Toc76737086)

[1. Introduction 7](#_Toc76737087)

[1.1. Motivations 8](#_Toc76737088)

[1.2. Objectives and Scope 8](#_Toc76737089)

[2. Design and Implementation 11](#_Toc76737090)

[2.1. Design 11](#_Toc76737091)

[2.1.1 ERD 11](#_Toc76737095)

[2.1.2 Use Cases 12](#_Toc76737096)

[2.1.3 Activity Diagram 13](#_Toc76737097)

[Voter side: 13](#_Toc76737098)

[Admin Side: 14](#_Toc76737099)

[2.1.4 Flow Diagram: 15](#_Toc76737100)

[2.1.5 Sequence Diagrams 20](#_Toc76737101)

[2.2 Results 23](#_Toc76737102)

[Overall Description 23](#_Toc76737103)

[2.2.1 Features Overview 26](#_Toc76737106)

[2.2.2 General Comments 32](#_Toc76737107)

[2.2. Literature Revieew/Conclusion And Future Work: 33](#_Toc76737108)

[References 34](#_Toc76737109)

[Appendix A – Development Environment 36](#_Toc76737110)

[ Operating Systems(s) 36](#_Toc76737111)

[o Windows 36](#_Toc76737112)

[Programming Language(s) 37](#_Toc76737113)

[Programming Paradigm 37](#_Toc76737114)

[Object-oriented programming 37](#_Toc76737115)

[Communication and Leadership 38](#_Toc76737116)

[Democratic Decentralized 38](#_Toc76737117)

[Appendix B – Tools and Software 39](#_Toc76737118)

[Appendix C – Code 40](#_Toc76737119)

[Appendix D – Team Members 58](#_Toc76737120)

[Annexure I – Copy of Approved FYP Proposal 59](#_Toc76737121)

[Annexure II – Software Planning & Costing 64](#_Toc76737122)

# 

# Abstract

This application provides is a new technique of casting votes using mobile phones to deploy an easy and flexible way of casting votes anytime and from anywhere.Electronic voting is often seen as a tool for making the electoral process more efficient and for increasing trust in its management. Properly implemented,  E-voting solutions can increase the security of the ballot, speed up the processing of results, and make voting easier. However, the challenges are considerable. If not carefully planned and designed, e-voting can undermine confidence in the whole electoral process. This policy paper outlines contextual factors that can influence the success of e-voting solutions and highlights the importance of taking these fully into account before choosing to introduce new voting technologies.

The main purpose of this study is to boost the turnout of voters. For this purpose, we have to view all the aspects responsible for the low turnout. Some people hesitate to vote due to weather conditions in different areas during the election, youngsters of age group 18 – 24 having no charm to cast the vote. People who are outside of their town/city don’t want to come to their area for just casting the votes due to the expenses and trouble of transportation. The same situation is also for those who are on duty during the election, they don’t have any interest to cast their vote during the job or they don’t have the facility to submit their vote.

A second purpose is to make it more difficult to commit fraud and cheating during an election.The application is specially developed for the government to cast their votes for any new policy regulation or issues. The issues or arguments are fed into the system by the admin. In a manual system, sometimes people are registered in more than one area and can thus cast the vote multiple times. By creating an online database covering the country it will be possible to eliminate the double casting of votes. In some areas, officials of the Election Commission themselves cast votes and after the end of the election, they adjust these votes from the voter’s list. Hence, they manipulate the result of the election. There is a need for a system that could reduce the authority of officials and could sustain the true nature of voting. It would also give people options to cast an empty vote if they don’t like to give the vote to any of the candidates.People in Pakistan are well aware of available online services and, from the last few years the use of the internet has increased. Youngsters havedeveloped an interest in using the internet for various purposes.After looking above factors, the decision for online voting is more natural than before.The expenditure of an election will be decreased and it will cover people of all ages. It will be a facility for the people who have different problems such as mentioned earlier.

# 

# Acknowledgments

First of all we would like to acknowledge our parents, all the teachers, institutes and mentors that have always supported us and helped us reach this level. Our most humble regards to the Director CoCIS, Khalid Khan, our Program Managers, Kashif Bashir our FYP head Sir usman Khan and the outstanding FYP committee members, Maam Samrina, Sir Zubair that have time and again guided us, tested us and clarified many problematic aspects of our work. Adding to the long list we are grateful to have had sir Muneeb as our advisor.

# Introduction

Electronic voting is often seen as a tool for making the electoral process more efficientand for increasing trust in itsmanagement.Properly implemented,  E-voting solutions can increase the security of the ballot, speed up the processing of results, and make voting easier

Web based voting allows the voter to vote from any place in state or out of state. The voter’s details passed which is used to detect his details from hash and save it as the first matchingpoint. The voter’s National identification card number is used to retrieve and return his saved data from the database of the Supreme Council elections which is passed to the same detection algorithm to detect cnic from it and save it as second matching point. algorithm to check wither they are identical or not. If the results of the matching or. If he has right to vote then a voting form is presented to him. Second level authentication is done using One Time Password (OTP) principle. The OTP principle emphasizes that each time the user tries to log on, thus improving the security. The result shows that the proposed algorithm capable of finding over 99% of the voters in database and allows their voter to vote in approximately 40 seconds.

## Motivations

The proposed e-voting system is based on the well-established e-voting approach. The system has been designed to support a voting application in the real world environment taking into account specific requirements such as privacy, eligibility, convenience, receiptfreeness and verifiability. The proposed system aims to achieve secure digital voting without compromising its usability. Within this context, the system is designed using a web-based interface to facilitate user engagement with measures such as finger printing to protect against double voting. With a clear need to administer the voters, constituencies and candidates for constituencies, a user-friendly administrator interface is implemented to enable ease of access. Furthermore, the system allows all voters equal rights of participation and develops a fair and healthy competition among all the candidates while keeping the anonymity of the voters preserved. The cryptographic hash of the transaction (ID) is emailed to the voter as a proof that the vote has been casted which may later on be tracked outside the premises of the constituency.

## Objectives and Scope

The generic requirements for a typical e-voting system have been defined. We present a brief description of each requirement along with an explanation of how the proposed system fulfils it.

Privacy - Keeping an individual’s vote secret

The system leverages cryptographic properties of blockchain to achieve privacy of a voter. More specifically, as voter is registered into the system, a voter hash is generated by blockchain which is the unique identifier of a voter into the blockchain, and is protected from misuse due to collision resistance property of the cryptographic hash. Due to this, the traceability of a vote is also non-trivial thereby protecting the voter when under duress.

#### Eligibility - Allowing only registered voters to vote, with each such voter voting only once

All eligible users are required to register using unique identifiers such as government-issued documents to assert their eligibility. In addition to this, our system implements strong authentication mechanism usingfinger printing technology to assert that only authorized voters can access the system. Furthermore, the use of biometrics also enables the system to protect against double voting.

#### Receipt Freeness - Voters should be unable to prove to a third party that they voted in a particular way

The proposed system enables a voter to vote as per their choice and creates a cryptographic hash for each such event (transaction). This is important to achieve verifiability i.e. to verify if a certain vote was included in the count. However, possession of this hash does not allow to extract information about the way voter has voted.

#### Convenience - Voters must be able to vote easily, and everyone who is eligible must be able to vote

The system has been implemented using a user friendly web based interface with the voting process requiring minimal input from the user. For instance, fingerprinting is implemented for authentication mechanism to avoid the requirement to remember username/passwords. Furthermore, the overall process is integrated which enables the user to interact with it in a seamless manner.

#### Verifiability - The ability to trust the vote tallying process

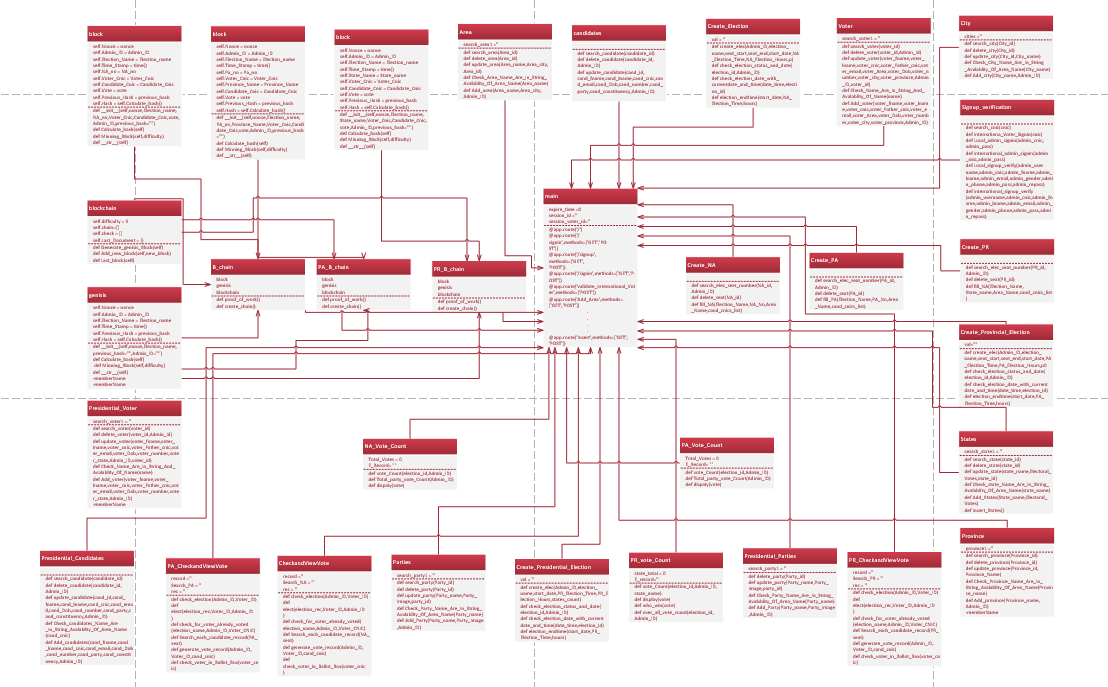
Upon casting their vote successfully, a user is provided with their unique transaction ID in the form of a cryptographic hash. A user can use this transaction ID to track if their vote was included in the tallying process. However, this process does not enable a user to view how they voted which has been adopted to mitigate threats when under duress. The analysis presented above highlights the performance of the proposed system with respect to the specific requirements of e-voting. It also highlights the significance of defining characteristics of blockchain and their profound role in achieving the cornerstones of an efficient e-voting system. Therefore, we believe the work presented here makes significant contribution to the existing knowledge with respect to the application of blockchain technology to achieve a secure digital voting system.

# Design and Implementation

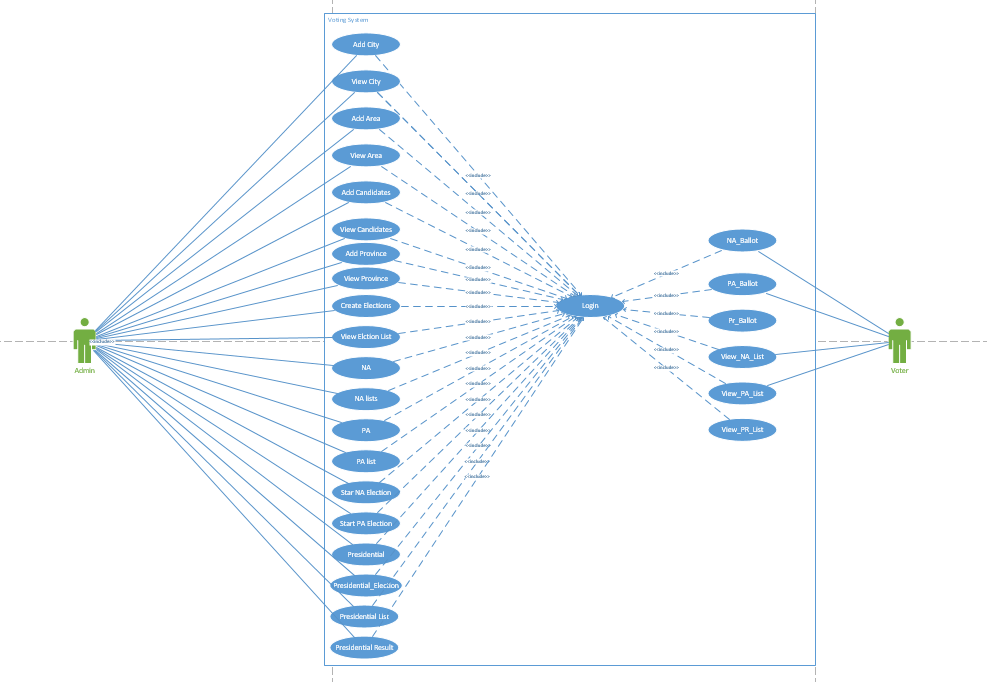
## Design



### ERD

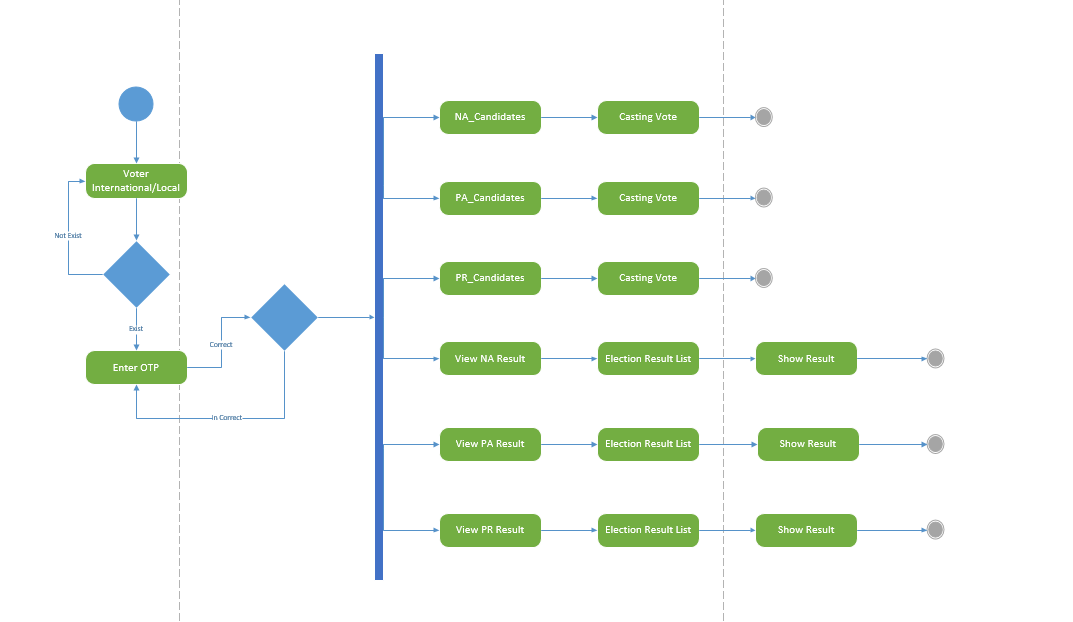


### Use Cases

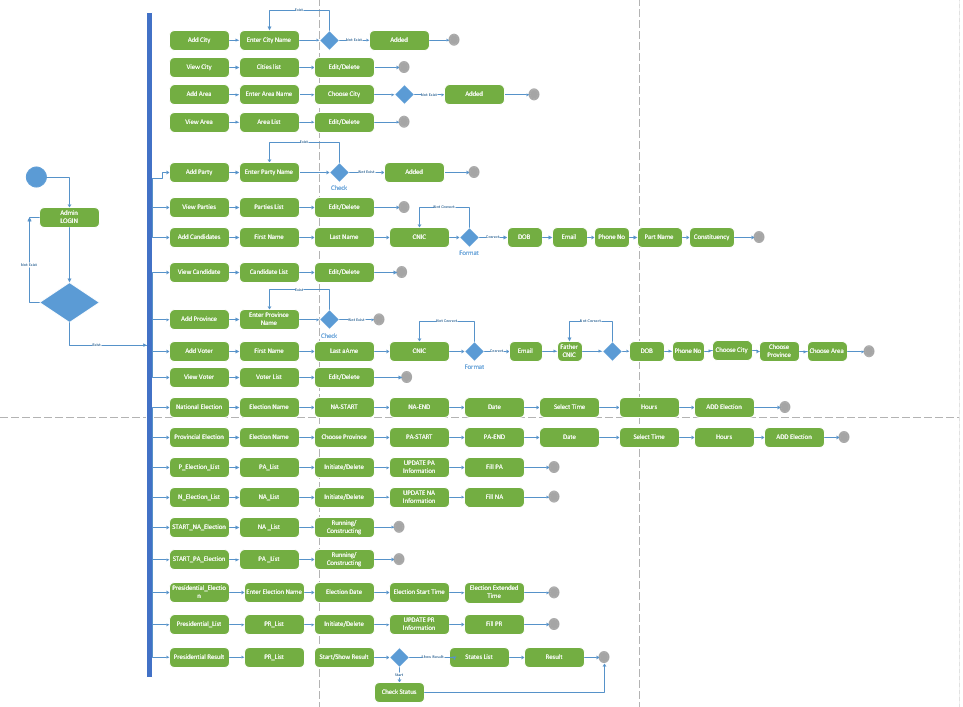


### Activity Diagram

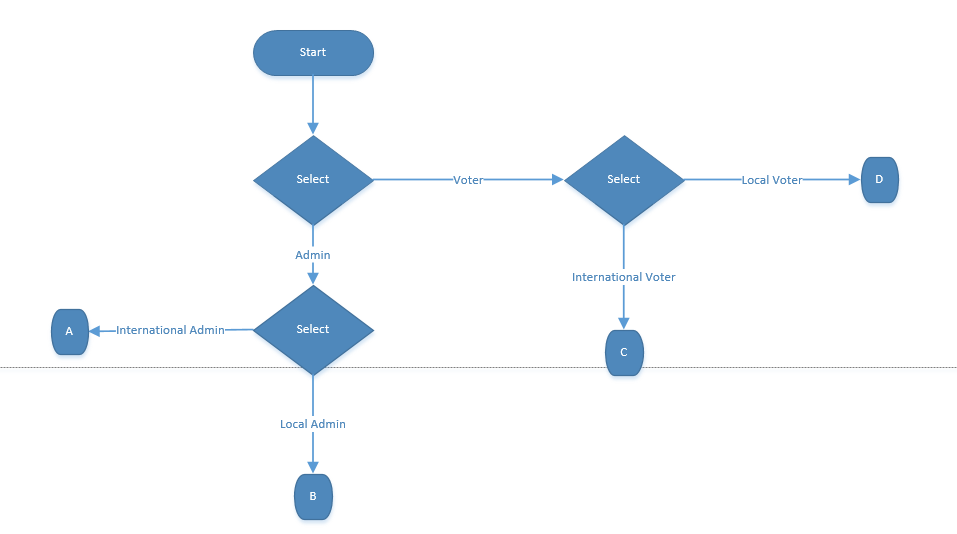
### Voter side:



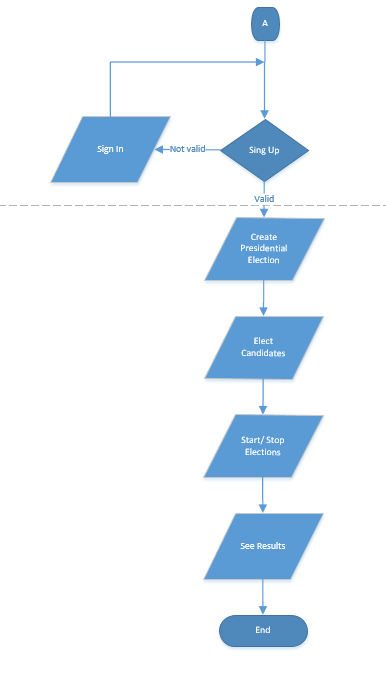
### Admin Side:



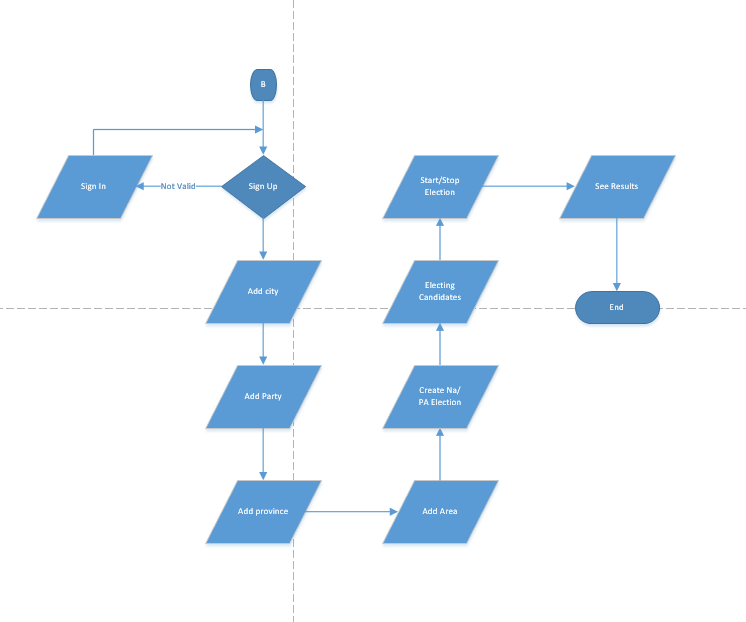
### Flow Diagram:



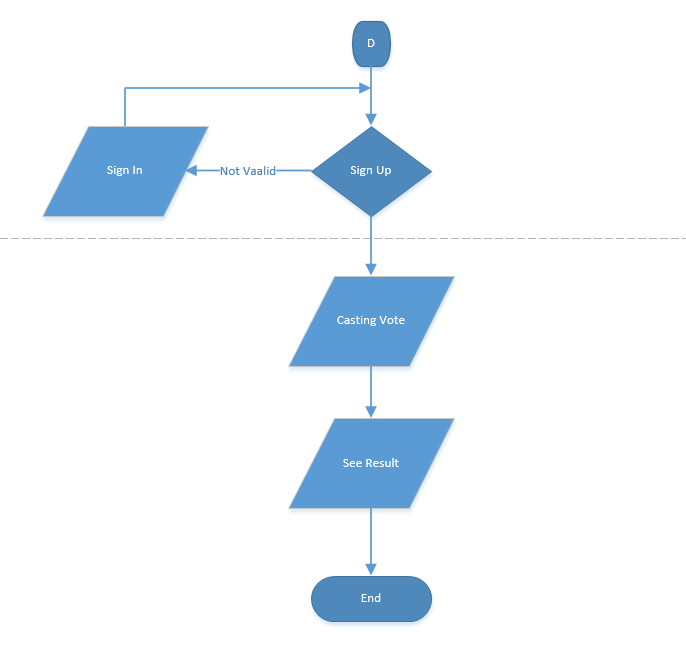
#### International Admin:



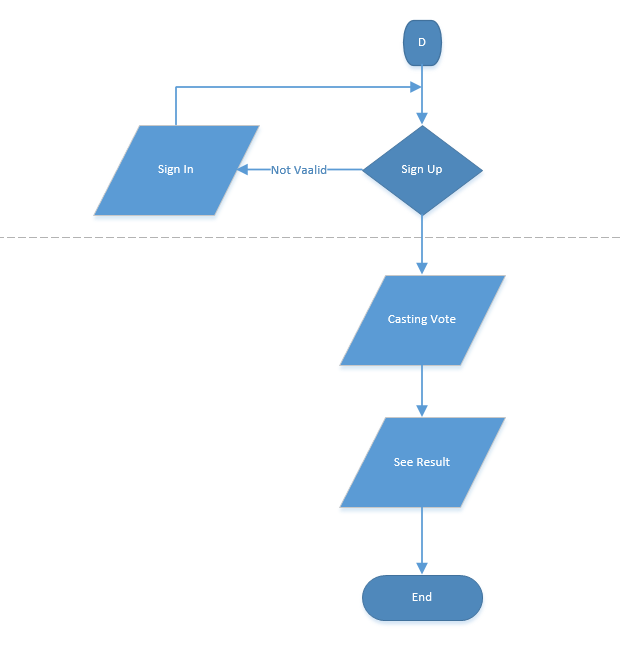
#### Local Admin:



#### International Voter:



#### Local Voter:

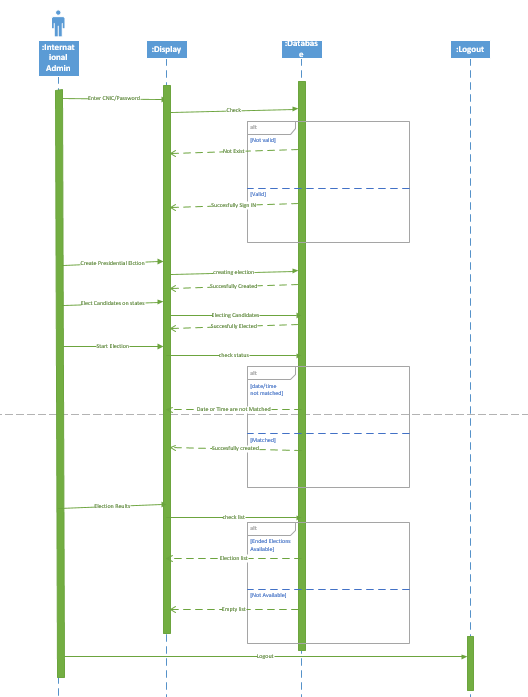


### Sequence Diagrams

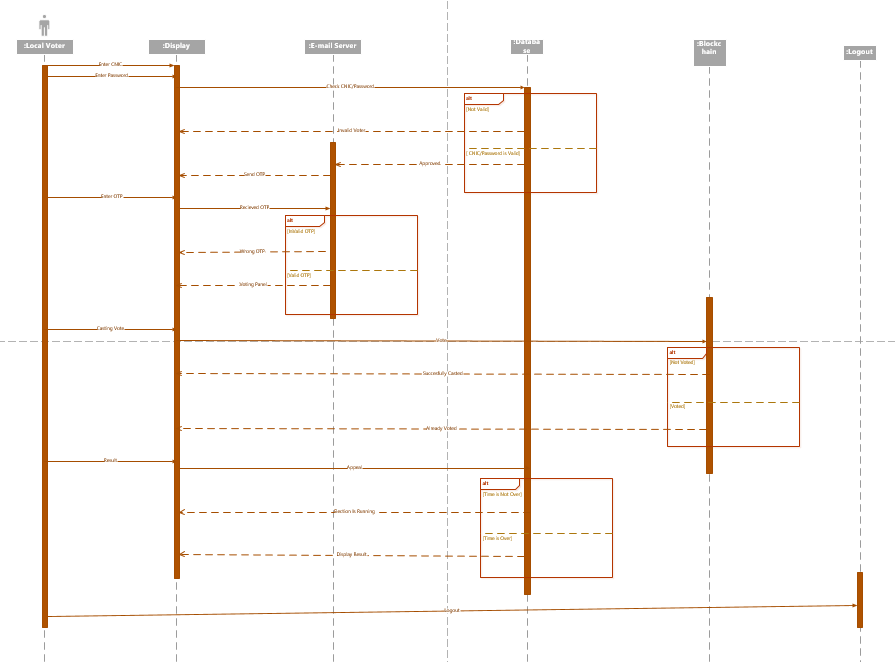
#### Local Admin:

### Sequence_Diagram_International_Admin

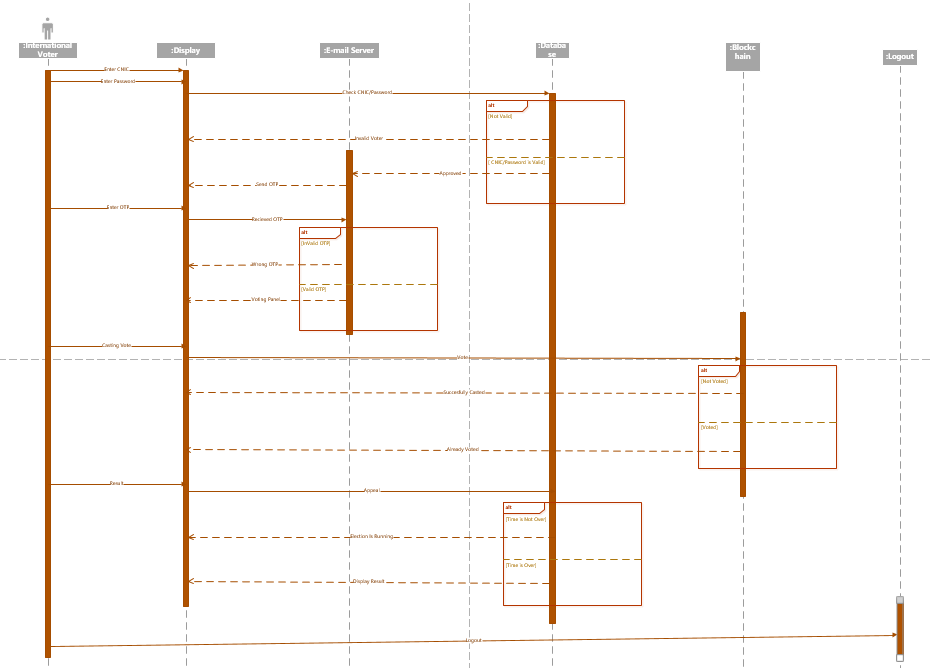
#### International Admin:



#### Local Voter:



#### International Voter:



## Results

### Overall Description

This software is being developed for use by everyone with a simple and self-explanatory GUI. This is software that can be used by people to vote in an election. All the user must do is log in and click on his favorable candidates to register his vote. The development and testing are done on Ethernet. While the online voting system has been an active area of research in recent years, the use of insecure Internet, well-documented cases of incorrect implementations reported recently. These challenges are to be resolved so that the public should cast their vote securely and conveniently. The proposed online voting system is a voting system by which any Voter can use his/her voting rights from anywhere in the country. The online voting system contains:

* Voter’s information in the database.
* Voter’s Names with ID and password.
* Voter’s vote in a database.
* Calculation of a total number of votes.

**Product Functions**

**PRODUCT REQUIREMENTS**

* Prepare the voting system for use in an election
* Produce the appropriate ballot formats
* Test that the voting system and ballot materials have been properly prepared and are ready for use
* Record and count votes
* Consolidate and report election results
* Display results on-site or remotely
* Produce and maintain comprehensive audit trail data
* Use nationally or internationally

**User Classes and Characteristics**

The product has a server backend that takes care of authenticating the users and maintaining necessary data structures. The GUI at the server's end enables creating the polls on behalf of the government. The users must connect to the server to authenticate their identification against the password and then vote using the GUI at their end. In this project, the government panel creates the election strand their candidates and users cast their vote. In the backend, the voter vote is connected with the blockchain this process secures the voter vote and the third person will never change their option.

**Software Platform**

Frontend : HTML,CSS,jinja2,javascript

Backend: Python flask

Database: MongoDB

**Scalability**

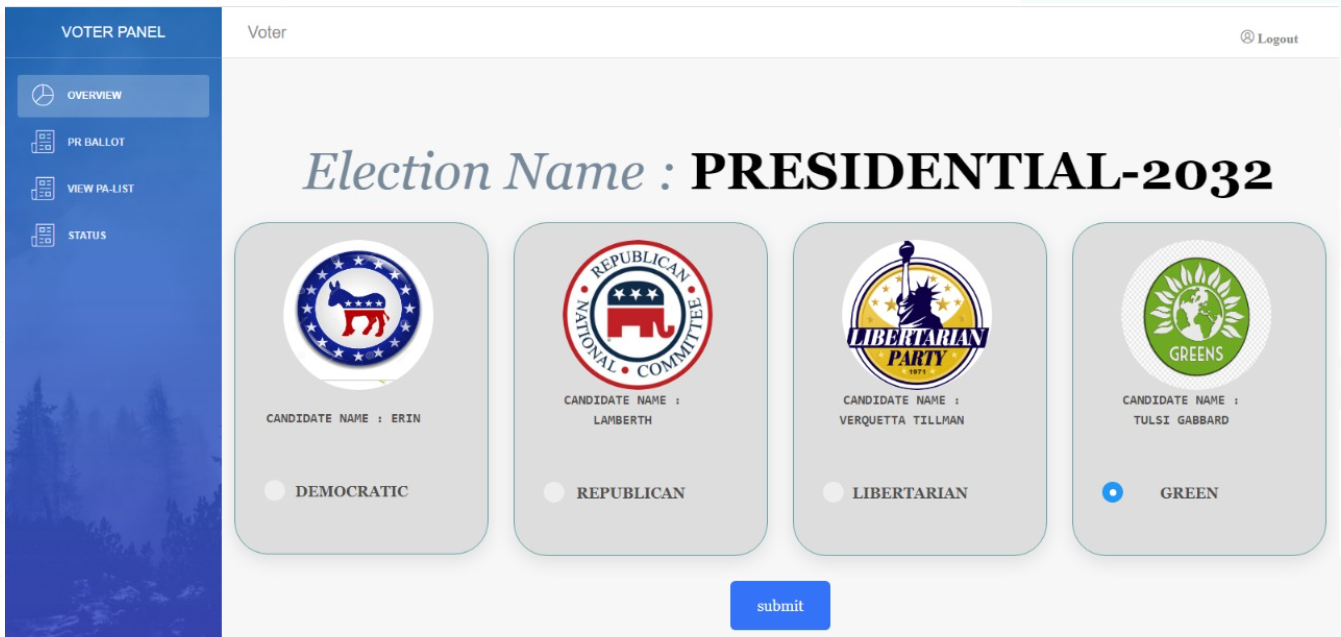
Users are considered to be technically novices but expected to be able to use a computer/hand-held terminal (HHT). and to click against the favorable candidate on the GUI



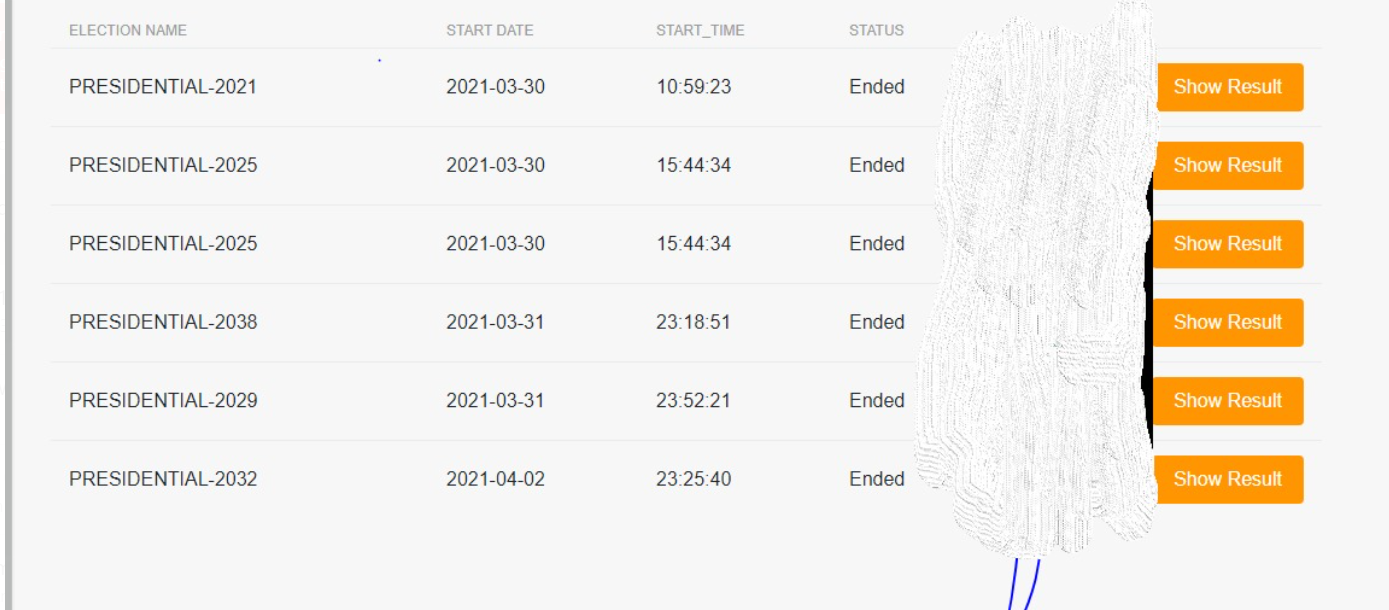
### Features Overview

The functionality and features included in the online voting system are detailed, illustrated and described below. Major features which have a high degree of inter-relatedness are grouped together as a sub-module where needed.

**National Election voter Panel :**

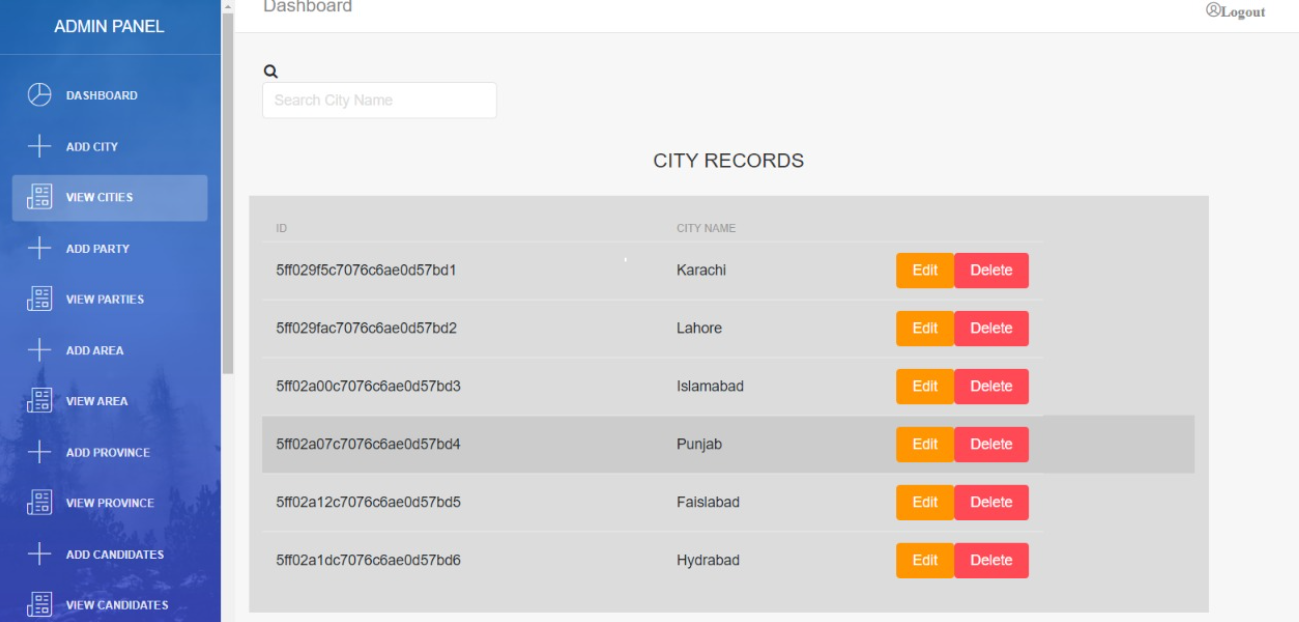
****

In the voter panel voter cast their vote at the backhand all this vote is connected with the blockchain through a cryptographic hash (hs-256). This will secure the voter vote and if the voter casts their vote the application will never allow casting again and if the voting starting and there is no candidate in their area so the application never allows casting their vote.

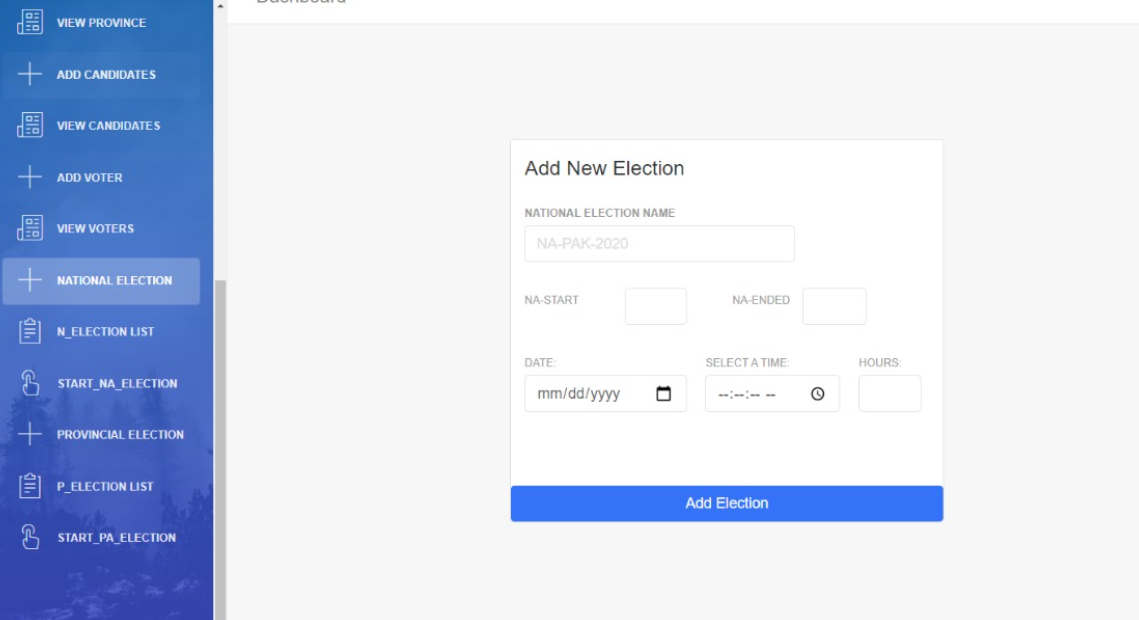


If the voting time ended the voter allow to check their results within a single click.

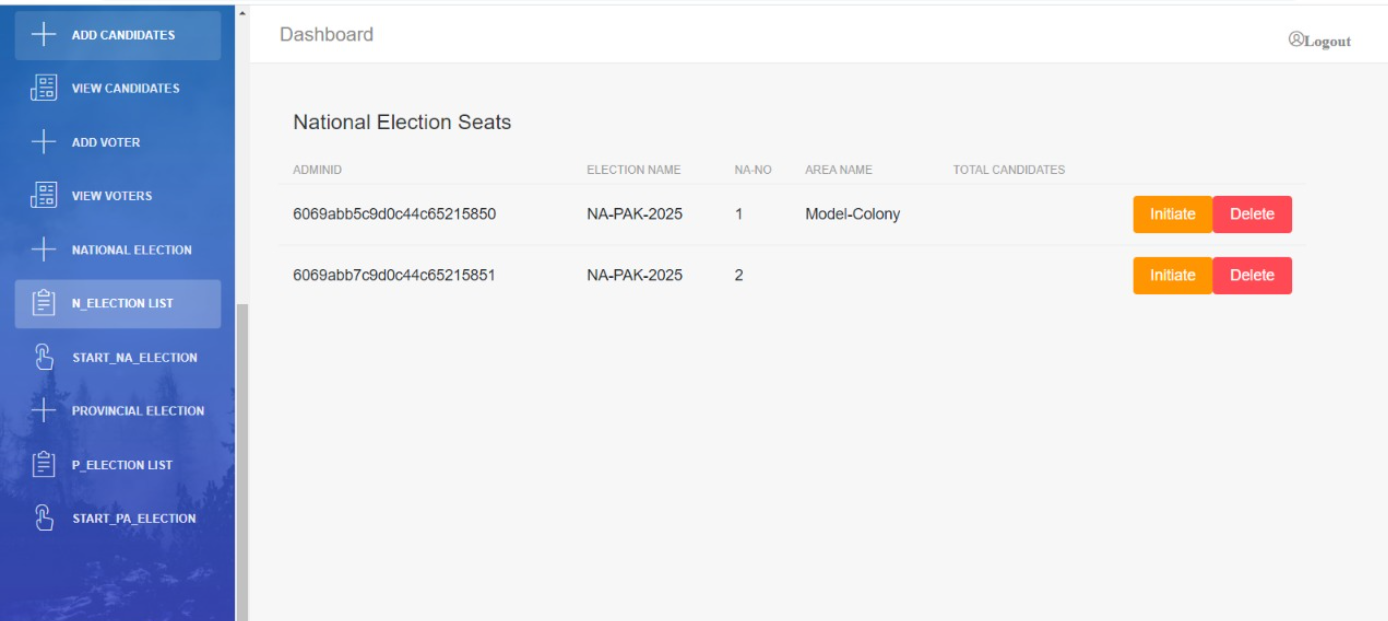
**National Election admin Panel:**



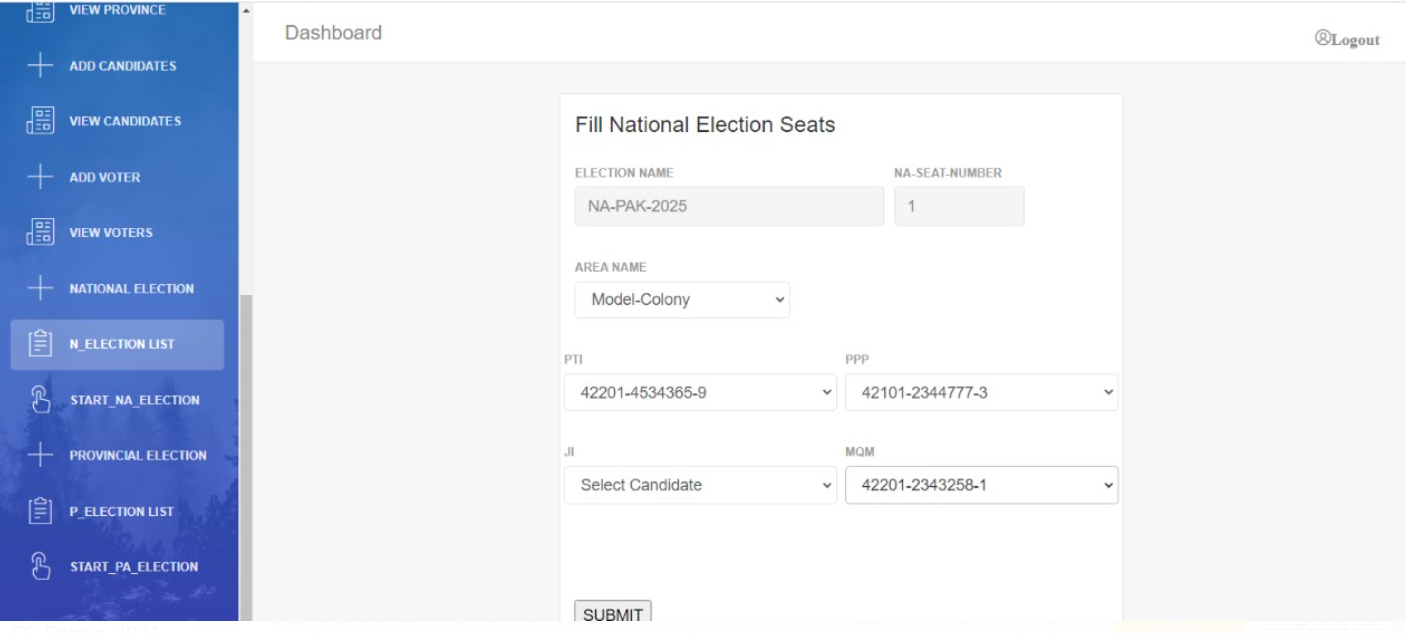
It admin panel works like a crud application admin can edit and delete the voting record in the cities.



In this admin can create an election assign the seat in the specific areas.



In this admin can set the election detail and assign the candidates.



In this national election list admin can set election name, seat number, area name and candidate.

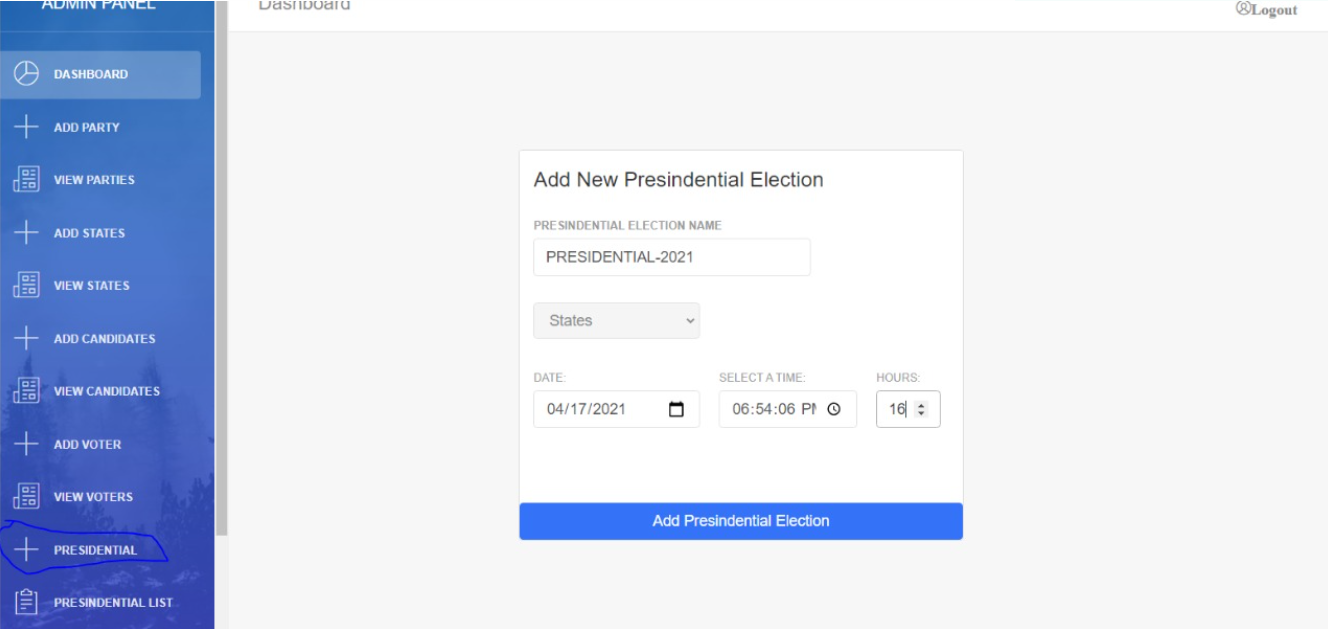


In this feature all election list show here it shows ended, running and constructing status in this admin can start to end the election and show the result

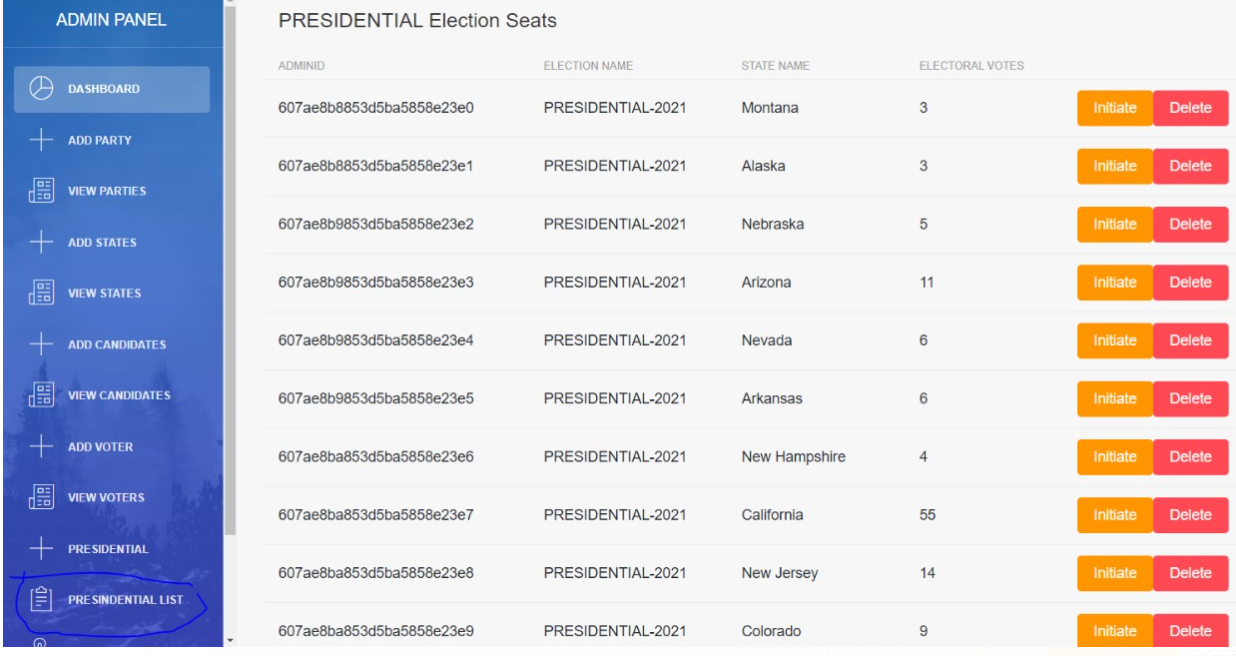
**International voting admin panel:**

****

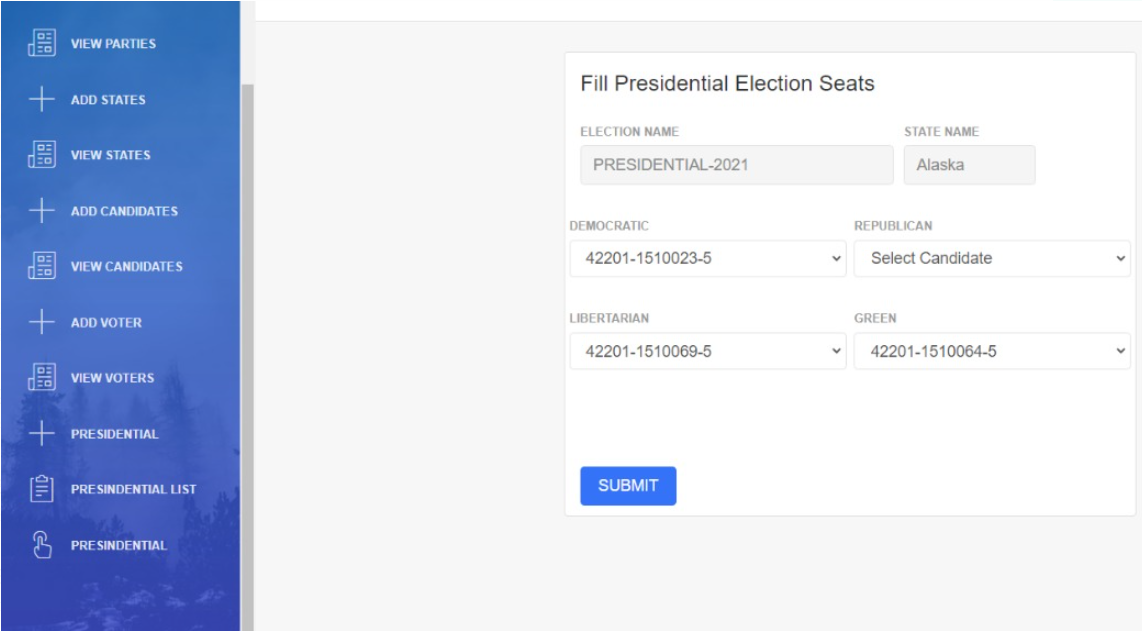
Admin panel for online voting system

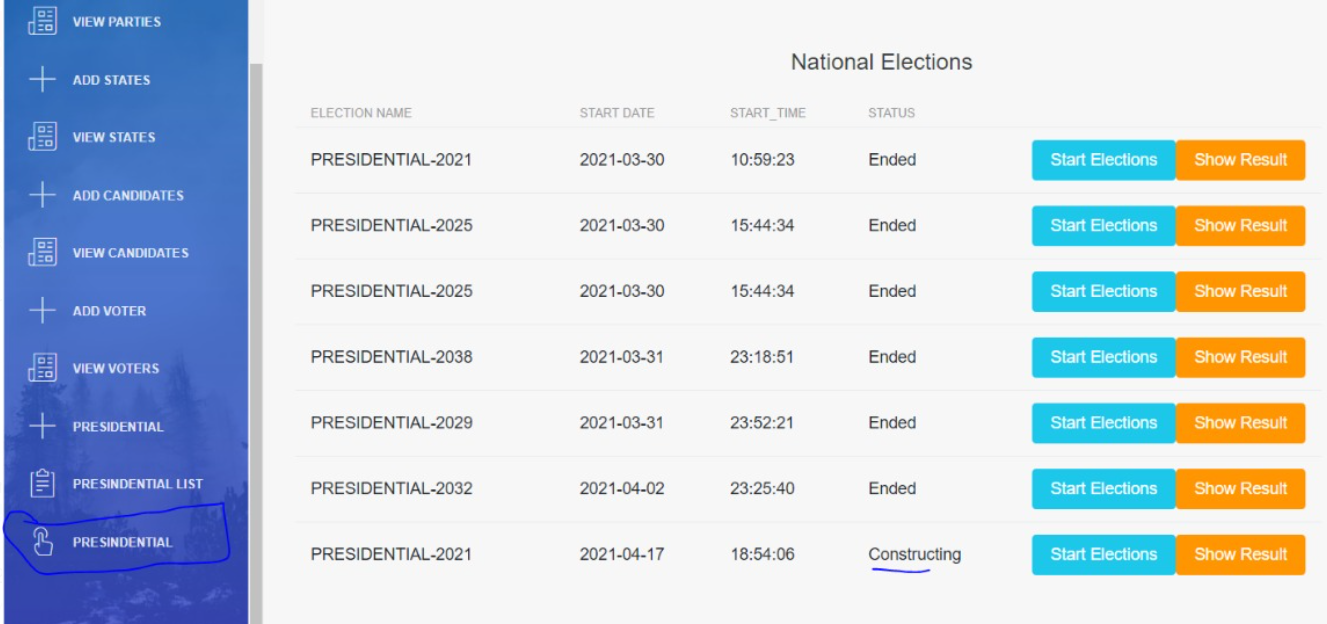


Add new election for international voting system



Show all the elected seats and votes in the application

In this admin create candidate option in the specific seats

Admin show all the ended and constructing election and show their results

### General Comments

As the features described above show, the requirements approved in the proposal have been implemented to the best of our efforts, the characteristics of the project combine to form a powerful environment that is user-friendly and provides an easy-to-use, intuitive Conclusion and Future Work

## Literature Revieew/Conclusion And Future Work:

**Literature Review:**

In (Kiayias & Yung, 2002), a self-tallying voting system is proposed that does not require any trusted third parties for vote aggregation and any private channel for voter-to-voter privacy. The proposed protocol involves extensive computation. In (Hao et al, 2010) a two round protocol is proposed that computes the tally in two rounds without using a private channel or a trusted third party. The protocol is efficient in terms of amputation and bandwidth consumption but is neither robust nor fair in certain conditions (Dalia et al, 2012). In (Dalia et al, 2012) a protocol is proposed to improve the robustness and fairness of the two round protocol (Hao et al, 2010). In (Shahandashti & Hao, 2016), authors propose E2E verifiable voting system named DRE-ip (DRE-i with enhanced privacy), that overcomes limitations of DRE-i (Chaum et al, 2008). Instead of pre-computing ciphertexts, DRE-ip encrypts the vote on the fly during voting process. DRE-ip achieves E2E verifiability without TAs, but at the same time provides a significantly stronger privacy guarantee than DRE-i. In (Chaum, 2004) end-to-end verifiability is achieved through the Mixnet protocol (Chaum, 1981) that recovers the plaintext ballot in an unlikable manner by randomizing the ciphertext through a chain of mix servers.

Scantegrity is proposed in (Chaum et al, 2008) that achieves end-to-end (E2E) verifiability with confirmation codes that allow voters to prove to themselves that their ballots are included in the final tally as they really are. Another scheme Prêt à Voter based on (Chaum, 2004) is proposed in (Chaum et al, 2005) that ensures privacy by constructing the ballot with two columns i.e. voting options are listed in one column and the voter's choice is entered in an adjacent column. The work in (Adida & Rivest, 2006) is based on Prêt à Voter but using homomorphic tabulation and it uses scratch stripes to allow off-line auditing of ballots. Other systems that have been proposed for electronic voting include: Bingo Voting (Bohli et al, 2007), Helios (Adida, 2008), DRE-i (Hao et al, 2014 ) and DRE-ip (Shahandashti & Hao, 2016), Star-Vote (Bell et al, 2013) and (Sandler et al, 2008) to name a few.

The existing approaches perform well for end-to-end verifiability without compromising the privacy of voters. In (McCorry et al, 2017), authors presented the implementation of decentralized and self-tallying internet voting protocol over the blockchain using Ethereum. Authors used the openvote (Chaum et al, 2008) e-voting approach as their baseline.

The focus of our research is to explore the exciting opportunities presented by blockchain technologies by investigating their application in diverse application domains. Within this context, this paper presents our efforts to develop an e-voting system by leveraging blockchain technology. To this end, our proposed scheme fulfils the specific requirements for e-voting as discussed in section 2 and illustrated further in the following sections.

**Conclusion And Future Work**

Electronic voting has been used in varying forms since 1970s with fundamental benefits over paper based systems such as increased efficiency and reduced errors. With the extraordinary growth in the use of blockchain technologies, a number of initiatives have been made to explore the feasibility of using blockchain to aid an effective solution to e-voting. This paper has presented one such effort which leverages benefits of blockchain such as cryptographic foundations and transparency to achieve an effective solution to e-voting. The proposed approach has been implemented with Multichain and in- depth evaluation of approach highlights its effectiveness with respect to achieving fundamental requirements for an e-voting scheme.

In continuation of this work, we are focused at improving the resistance of blockchain technology to ‘double spending’ problem which will translate as ‘double voting’ for e-voting systems. Although blockchain technology achieves significant success in detection of malleable change in a transaction however successful demonstration of such events have been achieve which motivates us to investigate it further. To this end, we believe an effective model to establish trustworthy provenance for e-voting systems will be crucial to achieve an end-to-end verifiable e-voting scheme. The work to achieve this is underway in the form of an additional provenance layer to aid the existing blockchain based infrastructure.

# References

* Adida, B.; ‘Helios (2008). Web-based open-audit voting, in Proceedings of the 17th Conference on Security Symposium, ser. SS'08. Berkeley, CA, USA: USENIX Association, 2008, pp. 335{348.
* Adida B. and Rivest, R. L. (2006). Scratch & vote: Self-contained paper-based cryptographic voting, in Proceedings of the 5th ACM Workshop on Privacy in Electronic Society, ser. WPES '06. New York, NY, USA: ACM, 2006, pp. 29-40.
* Bell, S., Benaloh, J., Byrne, M. D., Debeauvoir, D., Eakin, B., Kortum, P., McBurnett, N., Pereira, O., Stark, P. B., Wallach, D. S., Fisher, G., Montoya, J., Parker, M. and Winn, M. (2013). Star-vote: A secure, transparent, auditable, and reliable voting system, in 2013 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections (EVT/WOTE 13). Washington, D.C.: USENIX Association, 2013
* Bohli, J. M., Muller-Quade, J. and Rohrich, S. (2007). Bingo voting: Secure and coercion- free voting using a trusted random number generator, in Proceedings of the 1st International Conference on E-voting and Identity, ser. VOTE-ID'07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 111-124.
* Chaum, D., Essex, A., Carback, R., Clark, J., Popoveniuc, S., Sherman, A. and Vora, P. (2008) Scantegrity: End-to-end voter-veri\_able optical- scan voting, IEEE Security Privacy, vol. 6, no. 3, pp. 40- 46, May 2008.
* Chaum, D. (2004) Secret-ballot receipts: True voter-verifiable elections, IEEE Security Privacy, vol. 2, no. 1, pp. 38{47, Jan 2004.
* Chaum, D. (1981) Untraceable electronic mail, return addresses, and digital pseudonym’, Commun. ACM, vol. 24, no. 2, pp. 84{90, Feb. 1981.
* Chaum, D., Ryan, P. Y. A. and Schneider, P. Y. A. (2005). A practical voter-verifiable election scheme, in Proceedings of the 10th European Conference on Research in Computer Security, ser. ESORICS'05. Berlin, Heidelberg: Springer-Verlag, 2005, pp. 118- 139.
* Dalia, K., Ben, R. , Peter Y. A, and Feng, H. (2012) A fair and robust voting system by broadcast, 5th International Conference on E-voting, 2012.
* Kadam, M., Jha, P. Jaiswal, S. (2015) Double Spending Prevention in Bitcoins Network, International Journal of Computer Engineering and Applications, August 2015.
* Kiayias, A. and Yung, M. (2002) Self-tallying Elections and Perfect Ballot Secrecy. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 141{158.
* Kraft, D. (2015) Difficulty Control for Blockchain-Based Consensus System, Peer-to-Peer Networking and Applications by Springer, March 2015.
* McCorry, P., Shahandashti, S. F. and Hao. F. (2017) A smart contract for boardroom voting with maximum voter privacy in the proceedings of FC 2017.
* Multichain (2017) Open platform for blockchain applications. Available at: www.multichain.com last accessed: December 2017.
* Nakamoto., S. (2009) Bitcoin: A peer-to-peer electronic cash system, 2009 [Online]. Available:http://bitcoins.info/bitcoin-a-peer-to-peer-electroniccash-system-satoshi-nakamoto. Last accessed: December 2017.
* Narayanan, A., Bonneau, J., Felten, E., Miller, A. and Gold, S. (2015) Bitcoin and Cryptocurrency Technologies, Chapter 2 and 3, Draft October 2015.
* Rockwell, M. (2017) Bitcongress – Process for block voting and law, http://bitcongress.org/ last accessed: December 2017
* Rosenfeld. M. (2017). Analysis of hashrate-based double-spending. [Online]. Available: http://arxiv.org/abs/1402.2009 last accessed: December 2017.
* Rura L., Issac B., and Haldar M. K. (2016) Implementation and evaluation of steganography based online voting, International Journal of Electronic Government Research.
* Ryan, P. Y. A, (2008) Prêt à Voter with Paillier Encryption, in the Mathematical and Computer Modelling, in Vol. 48, issue 9-10,1646-1662, 2008.
* Shahandashti, F. S. and Hao, F. (2016) DRE-ip: A Verifiable E-Voting Scheme without Tallying Authorities, the 21st European Symposium on Research in Computer Security (ESORICS), 2016.
* Shahandashti S. F. and Hao, F. (2016). DRE-ip: A Verifiable E-Voting Scheme Without Tallying Authorities. Cham: Springer International Publishing, 2016, pp. 223-240.

# Appendix A – Development Environment

## Operating Systems(s)

### Windows

**Microsoft Windows** is a series of [graphical interface](http://en.wikipedia.org/wiki/Graphical_user_interface) [operating systems](http://en.wikipedia.org/wiki/Operating_system) developed, marketed, and sold by [Microsoft](http://en.wikipedia.org/wiki/Microsoft).

Microsoft introduced an [operating environment](http://en.wikipedia.org/wiki/Operating_environment) named *Windows* on November 20, 1985 as a graphical [operating system shell](http://en.wikipedia.org/wiki/Operating_system_shell) for [MS-DOS](http://en.wikipedia.org/wiki/MS-DOS) in response to the growing interest in [graphical user interfaces](http://en.wikipedia.org/wiki/Graphical_user_interface) (GUIs).[[5]](http://en.wikipedia.org/wiki/Microsoft_Windows#cite_note-aboutcomnov-5) Microsoft Windows came to [dominate](http://en.wikipedia.org/wiki/Dominance_(economics)) the world's [personal computer](http://en.wikipedia.org/wiki/Personal_computer) market with [over 90% market share](http://en.wikipedia.org/wiki/Usage_share_of_operating_systems), overtaking [Mac OS](http://en.wikipedia.org/wiki/Mac_OS), which had been introduced in 1984.

As of April 2014, the most recent versions of Windows for [personal computers](http://en.wikipedia.org/wiki/Personal_computer), [smartphones](http://en.wikipedia.org/wiki/Smartphone), [server computers](http://en.wikipedia.org/wiki/Server_(computing)) and [embedded devices](http://en.wikipedia.org/wiki/Embedded_system) are respectively [Windows 8.1](http://en.wikipedia.org/wiki/Windows_8.1),[Windows Phone 8.1](http://en.wikipedia.org/wiki/Windows_Phone_8.1), [Windows Server 2012 R2](http://en.wikipedia.org/wiki/Windows_Server_2012_R2) and [Windows Embedded 8](http://en.wikipedia.org/wiki/Windows_Embedded_8).

*Developer* – Microsoft  
*Version –* Windows 10

*Environment –* 64 Bit

*Website*–[windows.microsoft.com](https://apple.com/mac)

## Programming Language(s)

Python

**Python** is an [interpreted](https://en.wikipedia.org/wiki/Interpreted_language) [high-level](https://en.wikipedia.org/wiki/High-level_programming_language) [general-purpose programming language](https://en.wikipedia.org/wiki/General-purpose_programming_language). Python's design philosophy emphasizes [code readability](https://en.wikipedia.org/wiki/Code_readability) with its notable use of [significant indentation](https://en.wikipedia.org/wiki/Off-side_rule). Its [language constructs](https://en.wikipedia.org/wiki/Language_construct) as well as its [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) approach aim to help [programmers](https://en.wikipedia.org/wiki/Programmers) write clear, logical code for small and large-scale projects.

Python is [dynamically-typed](https://en.wikipedia.org/wiki/Dynamic_programming_language) and [garbage-collected](https://en.wikipedia.org/wiki/Garbage_collection_(computer_science)). It supports multiple [programming paradigms](https://en.wikipedia.org/wiki/Programming_paradigms), including [structured](https://en.wikipedia.org/wiki/Structured_programming) (particularly, [procedural](https://en.wikipedia.org/wiki/Procedural_programming)), [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) and [functional programming](https://en.wikipedia.org/wiki/Functional_programming). Python is often described as a "batteries included" language due to its comprehensive [standard library](https://en.wikipedia.org/wiki/Standard_library).

[Guido van Rossum](https://en.wikipedia.org/wiki/Guido_van_Rossum) began working on Python in the late 1980s, as a successor to the [ABC programming language](https://en.wikipedia.org/wiki/ABC_(programming_language)), and first released it in 1991 as Python 0.9.0.[]](https://en.wikipedia.org/wiki/Python_(programming_language)#cite_note-32) Python 2.0 was released in 2000 and introduced new features, such as [list comprehensions](https://en.wikipedia.org/wiki/List_comprehension) and a garbage collection system using [reference counting](https://en.wikipedia.org/wiki/Reference_counting) and was discontinued with version 2.7.18 in 2020. Python 3.0 was released in 2008 and was a major revision of the language that is not completely [backward-compatible](https://en.wikipedia.org/wiki/Backward_compatibility) and much Python 2 code does not run unmodified on Python 3.

Python consistently ranks as one of the most popular programming languages.

## Programming Paradigm

### Object-oriented programming

Object-oriented programming (OOP) is a programming paradigm that represents the concept of "objects" that have data fields (attributes that describe the object) and associated procedures known as methods. Objects, which are usually instances of classes, are used to interact with one another to design applications and computer programs.

**Object**-**oriented analysis and design**

Object Oriented Analysis & Design (OOAD) is a popular technical approach to analyzing, **designing** an application, system, or business by applying

the **object**-**oriented** paradigm and visual modeling throughout the development life cycles to foster better stakeholder communication and product quality.

## Communication and Leadership

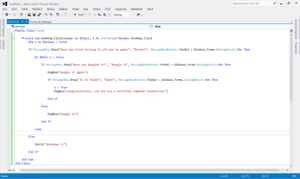
### Democratic Decentralized

The crucial delegation of power among group mates remains that power is equally distributed as well as the responsibility. We both had the veto power to decide upon the development processes and methodologies and the functions and modules were distributed while we worked separately on them with the consensus and co-operation of our partner. The power did not rest with any one particular individual and overall it was a constructive and to-and-fro development.

# Appendix B – Tools and Software

Microsoft Visual Studio 2010

[Visual Studio 2012 logo and wordmark.svg](http://en.wikipedia.org/wiki/File:Visual_Studio_2012_logo_and_wordmark.svg)

[](http://en.wikipedia.org/wiki/File:Visual_Studio_2012_EN.png)

**Microsoft Visual Studio** is an[integrated development environment](http://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) from [Microsoft](http://en.wikipedia.org/wiki/Microsoft). It is used to develop [computer programs](http://en.wikipedia.org/wiki/Computer_program) for [Microsoft Windows](http://en.wikipedia.org/wiki/Microsoft_Windows), as well as [web sites](http://en.wikipedia.org/wiki/Web_site), [web applications](http://en.wikipedia.org/wiki/Web_application) and [web services](http://en.wikipedia.org/wiki/Web_service).

Developers: Microsoft

Code Name: Dev10/Rosario

Supported .NET Framework versions: 2.0, 3.0, 3.5, 4.0

Release Date: April 12’ 2010

Internal Version: 10.0

Written in: C++ & C#

Type: Integrated Development Environment

License: Proprietary software

Express Edition: Registerware

Other editions: Trialware

Website: [www.visualstudio.com](http://www.visualstudio.com)

**MongoDB:**

MongoDB is a source-available cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas. MongoDB is developed by MongoDB Inc. and licensed under the Server Side Public License

[Stable release](https://www.google.com/search?sxsrf=ALeKk00eimfxW8UpIgaTCbw5neXgQwPVRA:1618686338780&q=mongodb+stable+release&sa=X&ved=2ahUKEwiBnuDB_IXwAhXSmFwKHYC3D-gQ6BMoADA0egQIMRAC): 4.4.4 / 16 February 2021，

[License](https://www.google.com/search?sxsrf=ALeKk00eimfxW8UpIgaTCbw5neXgQwPVRA:1618686338780&q=mongodb+license&sa=X&ved=2ahUKEwiBnuDB_IXwAhXSmFwKHYC3D-gQ6BMoADA1egQINRAC): [Server Side Public License](https://www.google.com/search?sxsrf=ALeKk00eimfxW8UpIgaTCbw5neXgQwPVRA:1618686338780&q=Server+Side+Public+License&stick=H4sIAAAAAAAAAONgVuLVT9c3NMyoSi7MzUuqXMQqFZxaVJZapBCcmZKqEFCalJOZrOCTmZyaV5wKAPGRok0uAAAA&sa=X&ved=2ahUKEwiBnuDB_IXwAhXSmFwKHYC3D-gQmxMoATA1egQINRAD)

[Initial release date](https://www.google.com/search?sxsrf=ALeKk00eimfxW8UpIgaTCbw5neXgQwPVRA:1618686338780&q=mongodb+initial+release+date&stick=H4sIAAAAAAAAAOPgE-LSz9U3MK2KLzLK01LOKLfST87PyUlNLsnMz9Mvzk8rKU8sSrVKLChITSxSSEksSV3EKpObn5een5KkkJmXWZKZmKNQlJqTmlicCpYGAERHK_hSAAAA&sa=X&ved=2ahUKEwiBnuDB_IXwAhXSmFwKHYC3D-gQ6BMoADA2egQINhAC): February 11, 2009

[Operating system](https://www.google.com/search?sxsrf=ALeKk00eimfxW8UpIgaTCbw5neXgQwPVRA:1618686338780&q=mongodb+operating+system&sa=X&ved=2ahUKEwiBnuDB_IXwAhXSmFwKHYC3D-gQ6BMoADA3egQIMhAC): [Windows Vista](https://www.google.com/search?sxsrf=ALeKk00eimfxW8UpIgaTCbw5neXgQwPVRA:1618686338780&q=Windows+Vista&stick=H4sIAAAAAAAAAONgVuLUz9U3MDc0SzJbxMobnpmXkl9erBCWWVySCAC4KuYVHQAAAA&sa=X&ved=2ahUKEwiBnuDB_IXwAhXSmFwKHYC3D-gQmxMoATA3egQIMhAD) and later, [Linux](https://www.google.com/search?sxsrf=ALeKk00eimfxW8UpIgaTCbw5neXgQwPVRA:1618686338780&q=Linux&stick=H4sIAAAAAAAAAONgVuLUz9U3SCuoqipYxMrqk5lXWgEATgerNhUAAAA&sa=X&ved=2ahUKEwiBnuDB_IXwAhXSmFwKHYC3D-gQmxMoAjA3egQIMhAE), [OS X](https://www.google.com/search?sxsrf=ALeKk00eimfxW8UpIgaTCbw5neXgQwPVRA:1618686338780&q=OS+X&stick=H4sIAAAAAAAAAONgVuLQz9U3MDWtLFrEyuIfrBABANwDvJoTAAAA&sa=X&ved=2ahUKEwiBnuDB_IXwAhXSmFwKHYC3D-gQmxMoAzA3egQIMhAF) 10.7 and later, [Solaris](https://www.google.com/search?sxsrf=ALeKk00eimfxW8UpIgaTCbw5neXgQwPVRA:1618686338780&q=Solaris&stick=H4sIAAAAAAAAAONgVuLQz9U3SM7IS1vEyh6cn5NYlFkMABylxwIWAAAA&sa=X&ved=2ahUKEwiBnuDB_IXwAhXSmFwKHYC3D-gQmxMoBDA3egQIMhAG), [FreeBSD](https://www.google.com/search?sxsrf=ALeKk00eimfxW8UpIgaTCbw5neXgQwPVRA:1618686338780&q=FreeBSD&stick=H4sIAAAAAAAAAONgVuLQz9U3MKpMqVjEyu5WlJrqFOwCAOg-RnYWAAAA&sa=X&ved=2ahUKEwiBnuDB_IXwAhXSmFwKHYC3D-gQmxMoBTA3egQIMhAH)

[Programming languages](https://www.google.com/search?sxsrf=ALeKk00eimfxW8UpIgaTCbw5neXgQwPVRA:1618686338780&q=mongodb+programming+languages&stick=H4sIAAAAAAAAAOPgE-LSz9U3MK2KLzLK09LJKLfST87PyUlNLsnMz9Mvzk8rKU8sSrUqKMpPL0rMzc3MS1fIScxLL01MT13EKpubn5een5KkgE26GABX5-m2XAAAAA&sa=X&ved=2ahUKEwiBnuDB_IXwAhXSmFwKHYC3D-gQ6BMoADA4egQIMBAC): [JavaScript](https://www.google.com/search?sxsrf=ALeKk00eimfxW8UpIgaTCbw5neXgQwPVRA:1618686338780&q=JavaScript&stick=H4sIAAAAAAAAAOPgE-LSz9U3MK2KLzLKU-IAsY0KLM21dDLKrfST83NyUpNLMvPz9Ivz00rKE4tSrQqK8tOLEnNzM_PSFXIS89JLE9NTF7FyeSWWJQYnF2UWlOxgZQQA1itDBlYAAAA&sa=X&ved=2ahUKEwiBnuDB_IXwAhXSmFwKHYC3D-gQmxMoATA4egQIMBAD), [Python](https://www.google.com/search?sxsrf=ALeKk00eimfxW8UpIgaTCbw5neXgQwPVRA:1618686338780&q=Python&stick=H4sIAAAAAAAAAOPgE-LSz9U3MK2KLzLKU-KAsA3jtXQyyq30k_NzclKTSzLz8_SL89NKyhOLUq0KivLTixJzczPz0hVyEvPSSxPTUxexsgVUlmTk5-1gZQQAtiAsClIAAAA&sa=X&ved=2ahUKEwiBnuDB_IXwAhXSmFwKHYC3D-gQmxMoAjA4egQIMBAE), [Java](https://www.google.com/search?sxsrf=ALeKk00eimfxW8UpIgaTCbw5neXgQwPVRA:1618686338780&q=Java&stick=H4sIAAAAAAAAAOPgE-LSz9U3MK2KLzLKUwKzzYuTstOStHQyyq30k_NzclKTSzLz8_SL89NKyhOLUq0KivLTixJzczPz0hVyEvPSSxPTUxexsnglliXuYGUEAAhvXZ9SAAAA&sa=X&ved=2ahUKEwiBnuDB_IXwAhXSmFwKHYC3D-gQmxMoAzA4egQIMBAF), [PHP](https://www.google.com/search?sxsrf=ALeKk00eimfxW8UpIgaTCbw5neXgQwPVRA:1618686338780&q=PHP&stick=H4sIAAAAAAAAAOPgE-LSz9U3MK2KLzLKU-IAsc0Mssu0dDLKrfST83NyUpNLMvPz9Ivz00rKE4tSrQqK8tOLEnNzM_PSFXIS89JLE9NTF7EyB3gE7GBlBADIsqfKTwAAAA&sa=X&ved=2ahUKEwiBnuDB_IXwAhXSmFwKHYC3D-gQmxMoBDA4egQIMBAG), [Go](https://www.google.com/search?sxsrf=ALeKk00eimfxW8UpIgaTCbw5neXgQwPVRA:1618686338780&q=Go+(programming+language)&stick=H4sIAAAAAAAAAOPgE-LSz9U3MK2KLzLKUwKzLdOTKrKKtHQyyq30k_NzclKTSzLz8_SL89NKyhOLUq0KivLTixJzczPz0hVyEvPSSxPTUxexSrrnK2hgk9LcwcoIAIlUy9pnAAAA&sa=X&ved=2ahUKEwiBnuDB_IXwAhXSmFwKHYC3D-gQmxMoBTA4egQIMBAH), [C++](https://www.google.com/search?sxsrf=ALeKk00eimfxW8UpIgaTCbw5neXgQwPVRA:1618686338780&q=C%2B%2B&stick=H4sIAAAAAAAAAOPgE-LSz9U3MK2KLzLKU-IAsbPSC9O1dDLKrfST83NyUpNLMvPz9Ivz00rKE4tSrQqK8tOLEnNzM_PSFXIS89JLE9NTF7EyO2tr72BlBADr1BjgTwAAAA&sa=X&ved=2ahUKEwiBnuDB_IXwAhXSmFwKHYC3D-gQmxMoBjA4egQIMBAI), [C](https://www.google.com/search?sxsrf=ALeKk00eimfxW8UpIgaTCbw5neXgQwPVRA:1618686338780&q=C&stick=H4sIAAAAAAAAAOPgE-LSz9U3MK2KLzLKU-IAsQ1LzJK0dDLKrfST83NyUpNLMvPz9Ivz00rKE4tSrQqK8tOLEnNzM_PSFXIS89JLE9NTF7EyOu9gZQQAgR48sk0AAAA&sa=X&ved=2ahUKEwiBnuDB_IXwAhXSmFwKHYC3D-gQmxMoBzA4egQIMBAJ), [Ruby](https://www.google.com/search?sxsrf=ALeKk00eimfxW8UpIgaTCbw5neXgQwPVRA:1618686338780&q=Ruby+programming+language&stick=H4sIAAAAAAAAAOPgE-LSz9U3MK2KLzLKU-IAsc3S0ky1dDLKrfST83NyUpNLMvPz9Ivz00rKE4tSrQqK8tOLEnNzM_PSFXIS89JLE9NTF7FKBpUmVSpgk9vByggAUFc_DmUAAAA&sa=X&ved=2ahUKEwiBnuDB_IXwAhXSmFwKHYC3D-gQmxMoCDA4egQIMBAK), [Perl](https://www.google.com/search?sxsrf=ALeKk00eimfxW8UpIgaTCbw5neXgQwPVRA:1618686338780&q=Perl&stick=H4sIAAAAAAAAAOPgE-LSz9U3MK2KLzLKU-KAsIvytHQyyq30k_NzclKTSzLz8_SL89NKyhOLUq0KivLTixJzczPz0hVyEvPSSxPTUxexsgSkFuXsYGUEAARNIItQAAAA&sa=X&ved=2ahUKEwiBnuDB_IXwAhXSmFwKHYC3D-gQmxMoCTA4egQIMBAL)

[Developer](https://www.google.com/search?sxsrf=ALeKk00eimfxW8UpIgaTCbw5neXgQwPVRA:1618686338780&q=mongodb+developer&stick=H4sIAAAAAAAAAOPgE-LSz9U3MK2KLzLK05LPTrbST87PLSgtSS3SL85PKylPLEq1SkktS83JL0gtWsQqmJufl56fkqQAFwMAg44M_kMAAAA&sa=X&ved=2ahUKEwiBnuDB_IXwAhXSmFwKHYC3D-gQ6BMoADA5egQILxAC): [MongoDB Inc.](https://www.google.com/search?sxsrf=ALeKk00eimfxW8UpIgaTCbw5neXgQwPVRA:1618686338780&q=MongoDB+Inc.&stick=H4sIAAAAAAAAAOPgE-LSz9U3MK2KLzLKUwKz0yvLzOPTtOSzk630k_NzC0pLUov0i_PTSsoTi1KtUlLLUnPyC1KLFrHy-Obnpee7OCl45iXr7WBlBADQtYyETQAAAA&sa=X&ved=2ahUKEwiBnuDB_IXwAhXSmFwKHYC3D-gQmxMoATA5egQILxAD), [Navionics](https://www.google.com/search?sxsrf=ALeKk00eimfxW8UpIgaTCbw5neXgQwPVRA:1618686338780&q=Navionics&stick=H4sIAAAAAAAAAOPgE-LSz9U3MK2KLzLKU-LVT9c3NEw2yo03qirI0pLPTrbST87PLSgtSS3SL85PKylPLEq1SkktS83JL0gtWsTK6ZdYlpmfl5lcvIOVEQBYApUTTQAAAA&sa=X&ved=2ahUKEwiBnuDB_IXwAhXSmFwKHYC3D-gQmxMoAjA5egQILxAE)

# Appendix C – Code

#### Blockchain for National Election:

import hashlib

import json

from time import time

from bson.objectid import ObjectId

from pymongo import MongoClient

import jwt

import random

# creation of MongoClient

client=MongoClient()

# Connect with the portnumber and host

client = MongoClient('mongodb+srv://saad:62908@cluster0.ojeri.mongodb.net/myFirstDatabase?authSource=admin&replicaSet=atlas-grm07z-shard-0&w=majority&readPreference=primary&retryWrites=true&ssl=true')

# Access database

db = client['votingsystem']

# Access collection of the database

ballot = db["Ballot"]

WB = db["Web\_Token"]

MB = db["Mined\_Block"]

# Genesis\_nounce = 0

# Remaining\_nouce = Genesis\_nounce

def proof\_of\_work():

    proof = random.randint(139587,99865127)

    return proof

class genisis:

    def \_\_init\_\_(self,nonce,Election\_name,previous\_hash="",Admin\_ID=""):

        self.Nonce = nonce

        self.Admin\_ID = Admin\_ID

        self.Election\_Name = Election\_name

        self.Time\_Stamp = time()

        self.Previous\_Hash = previous\_hash

        self.Hash = self.Calculate\_hash()

    def Calculate\_hash(self):

        block\_string = json.dumps({

            "Nonce": self.Nonce,

            "Admin\_ID": self.Admin\_ID,   #str(self.Admin\_ID)

            "Election\_name": self.Election\_Name,

            "Time\_Stamp":self.Time\_Stamp,

            "Previous\_Hash":self.Previous\_Hash

        },sort\_keys=True)

        Hash = hashlib.sha256(block\_string.encode()).hexdigest()

        return Hash

    def Minning\_Block(self,difficulty):

        while(self.Hash[:difficulty] != str('').zfill(difficulty)):                                     #Mining Block

            self.Nonce += 1

            self.Hash = self.Calculate\_hash()

        Insert\_to\_Mined\_Block = {

            "Blocked\_Mined": self.Hash,

        }

        MB.insert\_one(Insert\_to\_Mined\_Block)

    def \_\_str\_\_(self):

        Insert\_rec\_to\_Ballot = {

            "Admin\_ID": self.Admin\_ID,   #str(self.Admin\_ID)

            "Nonce": self.Nonce,

            "Election\_name": self.Election\_Name,

            "Time\_Stamp":self.Time\_Stamp,

            "Previous\_Hash":self.Previous\_Hash,

            "Hash": self.Hash

        }

        web\_token = jwt.encode(Insert\_rec\_to\_Ballot, "62908", algorithm="HS256")

        Insert\_web\_Token = {

            'Web\_Token' : web\_token

        }

        ballot.insert(Insert\_rec\_to\_Ballot)

        WB.insert(Insert\_web\_Token)

        string = "Previous\_Hash: " + str(self.Previous\_Hash) + "\n"

        string = string + "Hash: " + str(self.Hash) + "\n\n"

        string = string + "Web\_Token: " + str(web\_token) + "\n"

        return string

class block():

    def \_\_init\_\_(self,nonce,Election\_name,NA\_no,Voter\_Cnic,Candidate\_Cnic,vote,Admin\_ID,previous\_hash=""):

        self.Admin\_ID = Admin\_ID

        self.Nonce = nonce

        self.Time\_Stamp = time()

        self.Election\_Name = Election\_name

        self.NA\_no = NA\_no

        self.Voter\_Cnic = Voter\_Cnic

        self.Candidate\_Cnic = Candidate\_Cnic

        self.Vote = vote

        self.Previous\_Hash = previous\_hash

        self.Hash = self.Calculate\_hash()

    def Calculate\_hash(self):

        block\_string = json.dumps({

            "Admin\_ID": self.Admin\_ID,

            "Nonce":self.Nonce,

            "Time\_Stamp":self.Time\_Stamp,

            "Election\_name": self.Election\_Name,

            "NA\_no": self.NA\_no,

            "Voter\_CNIC": self.Voter\_Cnic,

            "Candidate\_Cnic":self.Candidate\_Cnic,

            "Vote":self.Vote,

            "Previous\_Hash":self.Previous\_Hash

        },sort\_keys=True)

        Hash = hashlib.sha256(block\_string.encode()).hexdigest()

        return Hash

    def Minning\_Block(self,difficulty):

        while(self.Hash[:difficulty] != str('').zfill(difficulty)):                                     #Mining Block

            self.Nonce += 1

            self.Hash = self.Calculate\_hash()

        Insert\_to\_Mined\_Block = {

            "Blocked\_Mined": self.Hash,

        }

        MB.insert(Insert\_to\_Mined\_Block)

    def \_\_str\_\_(self):

        Insert\_rec\_to\_Ballot = {

            "Admin\_ID": self.Admin\_ID,

            "Nonce":self.Nonce,

            "Time\_Stamp":self.Time\_Stamp,

            "Election\_name": self.Election\_Name,

            "NA\_no": self.NA\_no,

            "Voter\_CNIC": self.Voter\_Cnic,

            "Candidate\_Cnic":self.Candidate\_Cnic,

            "Vote":self.Vote,

            "Previous\_Hash":self.Previous\_Hash,

            "Hash": self.Hash

        }

        # print(Insert\_rec\_to\_Ballot)

        web\_token = jwt.encode(Insert\_rec\_to\_Ballot, "62908", algorithm="HS256")

        Insert\_web\_Token = {

            'Web\_Token' : web\_token

        }

        ballot.insert(Insert\_rec\_to\_Ballot)

        print('-------------------------Web-------------------------')

        print(web\_token)

        WB.insert(Insert\_web\_Token)

        string = "Previous\_Hash: " + str(self.Previous\_Hash) + "\n"

        string = string + "Hash: " + str(self.Hash) + "\n\n"

        string = string + "Web\_Token: " + str(web\_token) + "\n"

        return string

class Blockchain():

    def \_\_init\_\_(self):

        self.difficulty = 3

        self.chain=[]

        self.check = []

        self.Last\_Document = {}

        for i in ballot.find({}):

            self.check.append(i)

        if self.check == []:

            self.chain.append(self.Generate\_genisis\_Block())

        else:

            for i in range(0,len(self.check)):

                self.chain.append(self.check[i])

    def Generate\_genisis\_Block(self):

        genisis\_block = genisis(proof\_of\_work(),"NA-PAK-2020","GENISIS BLOCK")

        genisis\_block.Minning\_Block(self.difficulty)

        return genisis\_block

    def Last\_block(self):

       return self.chain[-1]

    def Add\_new\_block(self,new\_block):

        self.Last\_Document = self.Last\_block()

        if self.check == []:

            self.New\_Document = self.Last\_Document.Hash

        else:

            self.New\_Document = self.Last\_Document['Hash']

        new\_block.Previous\_Hash = self.New\_Document

        new\_block.Minning\_Block(self.difficulty)

        self.chain.append(new\_block)

def creat\_chain(block\_chain1):

    for i in block\_chain1.chain:

        print(i)

block\_chain = Blockchain()

block\_chain.Add\_new\_block(block(proof\_of\_work(),"NA-PAK-2020","1","PUNJAB","42201-1111111-5","42201-2222222-5","PTI","5ff0298cc7076c6ae0d57bd0"))

creat\_chain(block\_chain)

#### Blockchain For Provincial Election:

import hashlib

import json

from time import time

from bson.objectid import ObjectId

from pymongo import MongoClient

import jwt

import random

# creation of MongoClient

client=MongoClient()

# Connect with the portnumber and host

client = MongoClient('mongodb+srv://saad:62908@cluster0.ojeri.mongodb.net/myFirstDatabase?authSource=admin&replicaSet=atlas-grm07z-shard-0&w=majority&readPreference=primary&retryWrites=true&ssl=true')

# Access database

db = client['votingsystem']

# Access collection of the database

ballot = db["Provincial\_Ballot"]

WB = db["Web\_Token"]

MB = db["Mined\_Block"]

# Genesis\_nounce = 0

# Remaining\_nouce = Genesis\_nounce

def proof\_of\_work():

    proof = random.randint(139587,99865127)

    return proof

class genisis:

    def \_\_init\_\_(self,nonce,Election\_name,previous\_hash="",Admin\_ID=""):

        self.Nonce = nonce

        self.Admin\_ID = Admin\_ID

        self.Election\_Name = Election\_name

        self.Time\_Stamp = time()

        self.Previous\_Hash = previous\_hash

        self.Hash = self.Calculate\_hash()

    def Calculate\_hash(self):

        block\_string = json.dumps({

            "Nonce": self.Nonce,

            "Admin\_ID": self.Admin\_ID,   #str(self.Admin\_ID)

            "Election\_name": self.Election\_Name,

            "Time\_Stamp":self.Time\_Stamp,

            "Previous\_Hash":self.Previous\_Hash

        },sort\_keys=True)

        Hash = hashlib.sha256(block\_string.encode()).hexdigest()

        return Hash

    def Minning\_Block(self,difficulty):

        while(self.Hash[:difficulty] != str('').zfill(difficulty)):                                     #Mining Block

            self.Nonce += 1

            self.Hash = self.Calculate\_hash()

        Insert\_to\_Mined\_Block = {

            "Blocked\_Mined": self.Hash,

        }

        MB.insert\_one(Insert\_to\_Mined\_Block)

    def \_\_str\_\_(self):

        Insert\_rec\_to\_Ballot = {

            "Admin\_ID": self.Admin\_ID,   #str(self.Admin\_ID)

            "Nonce": self.Nonce,

            "Election\_name": self.Election\_Name,

            "Time\_Stamp":self.Time\_Stamp,

            "Previous\_Hash":self.Previous\_Hash,

            "Hash": self.Hash

        }

        web\_token = jwt.encode(Insert\_rec\_to\_Ballot, "62908", algorithm="HS256")

        Insert\_web\_Token = {

            'Web\_Token' : web\_token

        }

        ballot.insert(Insert\_rec\_to\_Ballot)

        WB.insert(Insert\_web\_Token)

        string = "Previous\_Hash: " + str(self.Previous\_Hash) + "\n"

        string = string + "Hash: " + str(self.Hash) + "\n\n"

        string = string + "Web\_Token: " + str(web\_token) + "\n"

        return string

class block():

    def \_\_init\_\_(self,nonce,Election\_name,Pa\_no,Province\_Name,Voter\_Cnic,Candidate\_Cnic,vote,Admin\_ID,previous\_hash=""):

        self.Admin\_ID = Admin\_ID

        self.Nonce = nonce

        self.Time\_Stamp = time()

        self.Election\_Name = Election\_name

        self.Pa\_no = Pa\_no

        self.Province\_Name = Province\_Name

        self.Voter\_Cnic = Voter\_Cnic

        self.Candidate\_Cnic = Candidate\_Cnic

        self.Vote = vote

        self.Previous\_Hash = previous\_hash

        self.Hash = self.Calculate\_hash()

    def Calculate\_hash(self):

        block\_string = json.dumps({

            "Admin\_ID": self.Admin\_ID,

            "Nonce":self.Nonce,

            "Time\_Stamp":self.Time\_Stamp,

            "Election\_name": self.Election\_Name,

            "Pa\_no": self.Pa\_no,

            "Province\_Name": self.Province\_Name,

            "Voter\_CNIC": self.Voter\_Cnic,

            "Candidate\_Cnic":self.Candidate\_Cnic,

            "Vote":self.Vote,

            "Previous\_Hash":self.Previous\_Hash

        },sort\_keys=True)

        Hash = hashlib.sha256(block\_string.encode()).hexdigest()

        return Hash

    def Minning\_Block(self,difficulty):

        while(self.Hash[:difficulty] != str('').zfill(difficulty)):                                     #Mining Block

            self.Nonce += 1

            self.Hash = self.Calculate\_hash()

        Insert\_to\_Mined\_Block = {

            "Blocked\_Mined": self.Hash,

        }

        MB.insert(Insert\_to\_Mined\_Block)

    def \_\_str\_\_(self):

        Insert\_rec\_to\_Ballot = {

            "Admin\_ID": self.Admin\_ID,

            "Nonce":self.Nonce,

            "Time\_Stamp":self.Time\_Stamp,

            "Election\_name": self.Election\_Name,

            "Pa\_no": self.Pa\_no,

            "Province\_Name": self.Province\_Name,

            "Voter\_CNIC": self.Voter\_Cnic,

            "Candidate\_Cnic":self.Candidate\_Cnic,

            "Vote":self.Vote,

            "Previous\_Hash":self.Previous\_Hash,

            "Hash": self.Hash

        }

        # print(Insert\_rec\_to\_Ballot)

        web\_token = jwt.encode(Insert\_rec\_to\_Ballot, "62908", algorithm="HS256")

        Insert\_web\_Token = {

            'Web\_Token' : web\_token

        }

        ballot.insert(Insert\_rec\_to\_Ballot)

        print('-------------------------Web-------------------------')

        print(web\_token)

        WB.insert(Insert\_web\_Token)

        string = "Previous\_Hash: " + str(self.Previous\_Hash) + "\n"

        string = string + "Hash: " + str(self.Hash) + "\n\n"

        string = string + "Web\_Token: " + str(web\_token) + "\n"

        return string

class Blockchain():

    def \_\_init\_\_(self):

        self.difficulty = 3

        self.chain=[]

        self.check = []

        self.Last\_Document = {}

        for i in ballot.find({}):

            self.check.append(i)

        if self.check == []:

            self.chain.append(self.Generate\_genisis\_Block())

        else:

            for i in range(0,len(self.check)):

                self.chain.append(self.check[i])

    def Generate\_genisis\_Block(self):

        genisis\_block = genisis(proof\_of\_work(),"PA-PAK-2020","GENISIS BLOCK")

        genisis\_block.Minning\_Block(self.difficulty)

        return genisis\_block

    def Last\_block(self):

       return self.chain[-1]

    def Add\_new\_block(self,new\_block):

        self.Last\_Document = self.Last\_block()

        if self.check == []:

            self.New\_Document = self.Last\_Document.Hash

        else:

            self.New\_Document = self.Last\_Document['Hash']

        new\_block.Previous\_Hash = self.New\_Document

        new\_block.Minning\_Block(self.difficulty)

        self.chain.append(new\_block)

def creat\_chain(block\_chain1):

    for i in block\_chain1.chain:

        print(i)

block\_chain = Blockchain()

block\_chain.Add\_new\_block(block(proof\_of\_work(),"NA-PAK-2020","1","PUNJAB","42201-1111111-5","42201-2222222-5","PTI","5ff0298cc7076c6ae0d57bd0"))

creat\_chain(block\_chain)

#### Blockchain For Presidential Election:

import hashlib

import json

from time import time

from bson.objectid import ObjectId

from pymongo import MongoClient

import jwt

import random

# creation of MongoClient

client=MongoClient()

# Connect with the portnumber and host

client = MongoClient('mongodb+srv://saad:62908@cluster0.ojeri.mongodb.net/myFirstDatabase?authSource=admin&replicaSet=atlas-grm07z-shard-0&w=majority&readPreference=primary&retryWrites=true&ssl=true')

# Access database

db = client['votingsystem']

# Access collection of the database

# ballot = db["Presidential\_Ballot"]

# h = db["Hash"]

MB = db["Presidential\_Mined\_Block"]

PB = db["Presidential\_Ballot"]

PWT = db["Presidential\_Ballot\_Web\_Token"]

PM = db["Presidential\_Mined\_Block"]

# Genesis\_nounce = 0

# Remaining\_nouce = Genesis\_nounce

def proof\_of\_work():

    proof = random.randint(139587,99865127)

    return proof

class genisis:

    def \_\_init\_\_(self,nonce,Election\_name,previous\_hash="",Admin\_ID=""):

        self.Nonce = nonce

        self.Admin\_ID = Admin\_ID

        self.Election\_Name = Election\_name

        self.Time\_Stamp = time()

        self.Previous\_Hash = previous\_hash

        self.Block\_Hash = self.Calculate\_hash()

    def Calculate\_hash(self):

        block\_string = json.dumps({

            "Nonce": self.Nonce,

            "Admin\_ID": self.Admin\_ID,   #str(self.Admin\_ID)

            "Election\_name": self.Election\_Name,

            "Time\_Stamp":self.Time\_Stamp,

            "Previous\_Hash":self.Previous\_Hash

        },sort\_keys=True)

        block\_hash = hashlib.sha256(block\_string.encode()).hexdigest()

        return block\_hash

    def Minning\_Block(self,difficulty):

        while(self.Block\_Hash[:difficulty] != str('').zfill(difficulty)):                                     #Mining Block

            self.Nonce += 1

            self.Block\_Hash = self.Calculate\_hash()

        Insert\_to\_Mined\_Block = {

            "Blocked\_Mined": self.Block\_Hash,

        }

        PM.insert\_one(Insert\_to\_Mined\_Block)

    def \_\_str\_\_(self):

        Insert\_rec\_to\_Ballot = {

            "Admin\_ID": self.Admin\_ID,   #str(self.Admin\_ID)

            "Nonce": self.Nonce,

            "Election\_name": self.Election\_Name,

            "Time\_Stamp":self.Time\_Stamp,

            "Previous\_Hash":self.Previous\_Hash,

            "Block\_Hash": self.Block\_Hash

        }

        web\_token = jwt.encode(Insert\_rec\_to\_Ballot, "62908", algorithm="HS256")

        Insert\_web\_Token = {

            'Web\_Token' : web\_token

        }

        PB.insert(Insert\_rec\_to\_Ballot)

        PWT.insert(Insert\_web\_Token)

        string = "Previous\_Hash: " + str(self.Previous\_Hash) + "\n"

        string = string + "Block\_hash: " + str(self.Block\_Hash) + "\n\n"

        string = string + "Web\_Token: " + str(web\_token) + "\n"

        return string

class block():

    def \_\_init\_\_(self,nonce,Election\_name,State\_name,Voter\_Cnic,Candidate\_Cnic,vote,Admin\_ID="",previous\_hash=""):

        self.Admin\_ID = Admin\_ID

        self.Nonce = nonce

        self.Time\_Stamp = time()

        self.Election\_Name = Election\_name

        self.State\_Name = State\_name

        self.Voter\_CNIC = Voter\_Cnic

        self.Candidate\_CNIC = Candidate\_Cnic

        self.Vote = vote

        self.Previous\_Hash = previous\_hash

        self.Block\_Hash = self.Calculate\_hash()

    def Calculate\_hash(self):

        block\_string = json.dumps({

            "Nonce": self.Nonce,

            "Admin\_ID": self.Admin\_ID,   #str(self.Admin\_ID)

            "Election\_name": self.Election\_Name,

            "Time\_Stamp":self.Time\_Stamp,

            "State\_Name":self.State\_Name,

            "Voter\_CNIC": self.Voter\_CNIC,

            "Candidate\_CNIC": self.Candidate\_CNIC,

            "Vote": self.Vote,

            "Previous\_Hash":self.Previous\_Hash

        },sort\_keys=True)

        block\_hash = hashlib.sha256(block\_string.encode()).hexdigest()

        return block\_hash

    def Minning\_Block(self,difficulty):

        while(self.Block\_Hash[:difficulty] != str('').zfill(difficulty)):                                     #Mining Block

            self.Nonce += 1

            self.Block\_Hash = self.Calculate\_hash()

        Insert\_to\_Mined\_Block = {

            "Blocked\_Mined": self.Block\_Hash,

        }

        PM.insert(Insert\_to\_Mined\_Block)

    def \_\_str\_\_(self):

        Insert\_rec\_to\_Ballot = {

            "Admin\_ID":  self.Admin\_ID,    #ObjectId(self.Admin\_ID)

            "Nonce": self.Nonce,

            "Election\_name": self.Election\_Name,

            "Time\_Stamp":self.Time\_Stamp,

            "State\_Name":self.State\_Name,

            "Voter\_CNIC": self.Voter\_CNIC,

            "Candidate\_CNIC": self.Candidate\_CNIC,

            "Vote": self.Vote,

            "Previous\_Hash":self.Previous\_Hash,

            "Block\_Hash": self.Block\_Hash

        }

        web\_token = jwt.encode(Insert\_rec\_to\_Ballot, "62908", algorithm="HS256")

        Insert\_web\_Token = {

            'Web\_Token' : web\_token

        }

        PB.insert(Insert\_rec\_to\_Ballot)

        PWT.insert(Insert\_web\_Token)

        string = "Previous\_Hash: " + str(self.Previous\_Hash) + "\n"

        string = string + "Block\_hash: " + str(self.Block\_Hash) + "\n\n"

        string = string + "Web\_Token: " + str(web\_token) + "\n"

        return string

class Blockchain():

    def \_\_init\_\_(self):

        self.difficulty = 3

        self.chain=[]

        self.check = []

        self.Last\_Document = {}

        for i in PB.find({}):

            self.check.append(i)

        if self.check == []:

            self.chain.append(self.Generate\_genisis\_Block())

        else:

            for i in range(0,len(self.check)):

                self.chain.append(self.check[i])

    def Generate\_genisis\_Block(self):

        genisis\_block = genisis(proof\_of\_work(),"PRESIDENTIAL-2020","GENISIS BLOCK")

        genisis\_block.Minning\_Block(self.difficulty)

        return genisis\_block

    def Last\_block(self):

       return self.chain[-1]

    def Add\_new\_block(self,new\_block):

        self.Last\_Document = self.Last\_block()

        if self.check == []:

            self.New\_Document = self.Last\_Document.Block\_Hash

        else:

            self.New\_Document = self.Last\_Document['Block\_Hash']

        new\_block.Previous\_Hash = self.New\_Document

        new\_block.Minning\_Block(self.difficulty)

        self.chain.append(new\_block)

def creat\_chain(block\_chain1):

    for i in block\_chain1.chain:

        print(i)

block\_chain = Blockchain()

block\_chain.Add\_new\_block(block(proof\_of\_work(),"Pak-2020","COlombia","42201-1111111-5","42201-2222222-5","Green"))

creat\_chain(block\_chain)

# Appendix D – Team Members

**Saad abbas**  (b. March 17, 1998) I am a student of Bachelors of Computer Science and I am currently in the final semester at PAF-KIET. Have furnished my skills in python, and mongodb.

Email: [saadabbaszulfiqar@gmail.com](mailto:saadabbaszulfiqar@gmail.com)

**Syed Ebadullah** (b. sep22, 1998) I am a student of Bachelors of Computer Science and I am currently in the final semester at PAF-KIET. Have furnished my skills in python, and MongoDB.

Email: ebadullah101@gmail.com

# Annexure I – Copy of Approved FYP Proposal

**FINAL YEAR PROJECT PROPOSAL   
[FALL 2020]**

**Summary Table**

|  |  |
| --- | --- |
| **Title of Project** | Online voting system |
| **Nature of End Product** | Web App |
| **Programming Environment** | 64-bit windows |
| **Programming Languages** | python / HTML5/ CSS/ |
| **Software Tools** | Visual studio  mongoDB |
| **Software Development Model** | Component Assembly Model / Water-fall / Agile |
| **Project Management Techniques** | Gantt charts/Program Evaluation and Review Technique / System Development Life Cycle |
| **Project Costing** | COCOMO / Cost & Benefit Analysis / Critical Success Factor |
| **Fault Tolerance** | Verification and Data Validation through White Box testing |
| **Team Members** | Saad abbas  (62908)  Syed ebadullah  (8979 ) |

**Accepted / Rejected:**­­ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ **Total Marks:**­­ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ **Committee Review:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
  
**Committee Signatures:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* 1. **Comparative survey of Past Work and Current Objective**

|  |  |  |
| --- | --- | --- |
| **#** | **Current Features** | **Proposed Features** |
| 01 | **Academic information** | **E-learning** |
| 02 | **Administrative information** | **Scalability of application – develop it so that it could run on mobiles and tablets** |
| 03 | **Updated solely by management** | **Introduce a platform for students to engage in conversations.** |

1. **Features / Challenges presented by the project**
   1. **Your project has ability to do this**

* National voting
* International voting
* Show result
* Admin panel
* User panel
  1. **Inputs / Data Collection methods**
* Voting
* Candidate info
  1. **Results and Reports**
  + Show voting result

**Nature of End-Product**

* 1. **Type:** It will be a web application capable of running on browsers All the information will be saved in databases uplinked to servers
  2. **Applicability / Impact on People’s life**: The product is aimed at all stakeholders within a government it improves the transparency

1. **Learning Out Comes**

Through this project, we are hoping to learn to handle various modules and databases and python

1. **Methodology**
   1. **Software Development:**  
      Project will be developed using Agile Development, specifically SCRUM
   2. **Code Pattern:**  
       Object Oriented Programming
2. **Testing Criteria**
   1. **Test Environment:** Automated and Manual Testing
   2. **Testing Approach:**White Box, Black Box
   3. **Testing Type:** Regression Testing, scalability testing, load testing, software performance testing, volume testing, reliability testing, compatibility testing, usability testing and graphical user interface testing.

# 

# Annexure II – Software Planning & Costing

**Project Budget & Costing**

* 1. **Budget &Costing Model**

The project is expected to cost at least PKR 40,000

* 1. **Revenue Generation**

The expenses would be split in two and the partners would bear them at their own discretion.

* 1. **Project Sponsor**

There are no current sponsors

1. **Hardware &Software Requirements**
   1. **Hardware**
      1. Android
      2. Tablet
      3. General PC
   2. **Software**
      1. MICROSOFT Visual Studio
      2. MongoDB Atlas
      3. Google Chrome