JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue

JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)

An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Automated Voting System

DrJ.Narendra Babu[1], Abhay Kumar Gowda[2], Amrutha Hosamani[3], Anushka Anand Shindhe[4], Ayush Kumar Shrivatsava[5],

[6] Arya K Chaitanya B T[7]

[1]drjnbabucse@gmail.com

[2,3,4,5,6,7] Students, Department of Data Science, Sapthagiri NPS University, India

[1]Professor, Department of Data Science, Sapthagiri NPS University, India

ABSTRACT

In modern society, ensuring secure and efficient voting is crucial for fair elections. Traditional method often faces issue such as errors and time consumption. To address this, the Automatic voting system is designed, to enhance accuracy and efficiency in voting. This system allows a predefined number of voters to vote for available candidates using unique voter ID.The main objective of this project is to design and develop a user-friendly system that allows voters to cast their votes electronically while maintaining data confidentiality and system security. To prevent duplicate voting, the program stores all submitted Voter IDs using a HashSet data structure, ensuring that each voter can vote only once. If a voter attempts to vote again, the system detects the duplicate and rejects the entry. Based on the user's input, votes are counted through conditional statements and switch-case logic. Each vote is recorded and stored securely in memory using simple counter variables. After all voters have cast their votes, the system automatically calculates and displays the total votes secured by each candidate. Finally, the program compares the vote counts and declares the winner, or indicates a tie if no candidate has a clear majority.This system uses basic Java concepts such as loops, conditional statements, switch-case blocks, and data structures. This system shows that it successfully records, counts, and displays results automatically, and demonstrates how automation can strengthen the election process by providing efficient voting mechanism suitable for educational institutions and small-scale elections.

Index Terms: Voting, Automated voting system, Voter ID, HashSet, Hashmap, ArrayList, Automation, Console-based.

I. INTRODUCTION

This program demonstrates an Automatic Voting System using Java collections such as HashSet, ArrayList, and HashMap. It allows the administrator to add candidates and registered voters, ensures that only valid voters can vote, and prevents duplicate voting. Each vote is counted and stored, and after the voting process ends, the program displays the total votes for each candidate and announces the winner.

Objectives

The goal of this project is to create an efficient and reliable automatic voting system. The specific goals include:

- To store and manage unique voters using HashSet and ensure no duplicate registrations occur.
- To prevent multiple voting attempts by tracking voters who have already voted.
- To maintain a list of candidates using ArrayList and record votes using HashMap.
- To ensure a secure, transparent, and error-free voting process.

II. LITERATURE SURVEY

Electronic voting has been widely studied as an effective method to automate and simplify the election process. Research in e-voting systems highlights the importance of secure voter authentication, prevention of duplicate voting, accurate counting mechanisms, and transparent result generation. Many academic prototypes and classroom systems use simple data structures to represent voters, candidates, and votes, as these offer a clear demonstration of the voting workflow without requiring complex infrastructure.

Existing literature on small-scale voting applications shows that HashSet is commonly used to store unique voters and prevent duplicates, while ArrayList helps maintain an ordered list of candidates for user-friendly display. HashMap is also frequently referenced as an effective structure for mapping candidates to their vote counts, allowing constant-time updates during voting. These structures together form the foundation of many introductory voting system models, demonstrating how basic data structures can support essential functionalities such as validation, input handling, and vote tallying.

Overall, previous studies support the design choices used in this project, as Java collections offer an efficient and straightforward way to implement a functional voting system. The reviewed literature shows that while this program represents a basic model, it successfully incorporates core components of electronic voting—voter verification, prevention of multiple votes, vote recording, and result calculation—which align with academic discussions on building reliable and transparent automated voting solutions.

III. PROPOSED METHOD

3.1 Software Architecture

The Automatic Voting System is developed using the Java programming language, and therefore requires a suitable software environment capable of compiling and executing Java programs. The project runs on the Java Development Kit (JDK), preferably version 8 or above, which provides essential tools such as the Java compiler and runtime environment. For coding and execution, an Integrated Development Environment (IDE) like Eclipse, is being used in this project for easier debugging and user-friendly interaction. The program relies on standard Java libraries and Java Collections to manage data efficiently using HashSet, ArrayList, and HashMap. Since the system is console-based, no additional frameworks, databases, or external tools are required. Any operating system such as Windows, Linux, or macOS that supports JDK and Eclipsecan run the project smoothly.

3.2OOP Concepts Implemented

According to the project abstract, the application applies the fundamental concepts of OOP, they include:

- Encapsulation: Data such as voters, candidates, and vote counts are stored inside the class as static variables.Access to data is controlled through methods inside the program logic rather than direct modification.
- Abstraction:The complex vote validation and counting operations are hidden from the user. The user interacts only with a simple input interface to cast votes.
- Inheritance and Polymorphism are not used here since the program is a single class solution and does not involve parent-child relationships or method overriding.

3.3 Modules Description

- Candidate Registration :This system allows the admin to enter the names of all candidates participating and stored in an ArrayList, and their initial vote count is set to zero in a HashMap.
- Voter Registration : This module collects and stores unique voter IDs using a HashSet and ensures that no voter is registered again.
- Voting Process :Voters enter their voter ID to cast their vote. The system checks if the ID is valid and ensures the voter has not already voted. The available candidates are displayed, and the voter selects one. The selected candidate's vote count is then updated in the HashMap.
- Winner Declaration : After all votes are cast, the system calculates the total votes received by each candidate and identifies the one with the highest vote count. This candidate is declared the winner, and the results are displayed.

IV. RESULTS

- The system successfully registered all candidates and initialized their vote counts.
- Unique voter IDs were stored correctly using HashSet, preventing duplicate registrations.
- The system validated each voter ID before allowing them to vote.
- Duplicate voting was prevented by maintaining a separate HashSet of voters who had already voted.
- The voting process recorded each vote accurately using a HashMap.
- The system displayed the total votes received by each candidate after voting ended.
- The candidate with the highest vote count was correctly identified and declared as the winner.
- Overall, the program ran efficiently and produced accurate, error-free results.

I. PROPOSED METHOD

3.1 System Architecture

The proposed method focuses on building a simple yet flexible currency conversion system using Java. The program stores exchange rates in a HashMap, with each currency mapped to its value relative to USD. This allows fast lookups and makes updates easy when users want to modify or add new rates

The User Interface Layer handles all user interactions via a console menu utilizing the Scanner class directing the user to select an operation.

The Application Logic Layer includes the functions that handle converting data checking input validity showing exchange rates and modifying values.

Data Storage Layer uses a HashMap to store and retrieve currency rates efficiently

3.2 OOP Concepts Implemented

The code utilizes core Object-Oriented Programming (OOP) principles to guarantee clarity, modular design and reusability. It employs Encapsulation by organizing currency-related actions—like conversion, updating rates and display features—within methods concealing the internal workings from the user and making them available only via controlled access points. Abstraction is achieved by breaking down operations such as conversion, into simple functions enabling users to engage with the system without requiring knowledge of the detailed computations. The code also demonstrates Modularity, a practical extension of OOP principles, by organizing different functionalities into separate methods, ensuring cleaner structure and easier maintenance.

3.3 Modules Description

- Menu Module: Shows all choices. Manages the overall program sequence by guiding the user to the chosen task.
- Input Handling Module: Collects user inputs such as currency codes, amounts, and updated rates using the Scanner class.
- Conversion Module: Executes the currency exchange by changing the source amount into USD and subsequently into the desired currency.
- Currency Exchange Display Module: Presents all currencies together, with their prevailing exchange rates.
- Rates Update Module: Enables users to change currency rates or insert new currency records.
- Data Storage Module: Uses a HashMap to store and retrieve all currency codes and their exchange values efficiently..

.IV. RESULTS

- The application effectively transforms currency amounts utilizing exchange rates kept within a HashMap.
- Users have the ability to input any value and choose source and target currencies, with the system showing the converted result.
- The two-stage conversion process (source → USD → target) yielded precise outcomes throughout the testing phase.
- The menu-based interface functioned seamlessly enabling users to select conversion check rates or modify options.
- Upon updating the exchange rates the program instantly implemented the figures without needing a restart.
- Every supported currency reacted appropriately. The software managed valid user entries efficiently.
- Overall, the results show the system is functional, reliable, and user-friendly for basic currency conversion tasks.

```
===== AUTOMATIC VOTING SYSTEM =====
Enter number of candidates: 6
Enter candidate name: Krishna
Enter candidate name: Ram
Enter candidate name: Rahul
Enter candidate name: Shri
Enter candidate name: Hari
Enter candidate name: Keshav
```

1. CANDIDATE REGISTRATION

```
Enter number of voters: 10
Enter voter ID: 001
Enter voter ID: 005
Enter voter ID: 007
Enter voter ID: 012
Enter voter ID: 014
Enter voter ID: 019
Enter voter ID: 027
Enter voter ID: 028
Enter voter ID: 029
Enter voter ID: 030
```

2.VOTER REGISTRATION

```
Enter voter ID (or type EXIT to stop voting): 005
Select a candidate:
1. Krishna
2. Ram
3. Rahul
4. Shri
5. Hari
6. Keshav
Enter your choice: 2
Vote cast for Ram
```

3. VOTIG PROCESS

```
Enter voter ID (or type EXIT to stop voting): 003
Invalid Voter ID.

Enter voter ID (or type EXIT to stop voting): 004
Invalid Voter ID.
```

4. INVALID VOTIG PREVENTION

```
Enter voter ID (or type EXIT to stop voting): 005
You have already voted!
```

5. DOUBLE VOTIG PREVENTION

```
Enter voter ID (or type EXIT to stop voting): exit
```

6.EXIT

```
===== Voting Results =====
Krishna → 1 votes
Ram → 1 votes
Rahul → 2 votes
Shri → 0 votes
Hari → 1 votes
Keshav → 5 votes

 Winner: Keshav with 5 votes!
```

7.WINNER DECLARATION

V. CONCLUSION

The development of the Automatic Voting System using Java demonstrates a structured and efficient approach to managing the essential components of an electronic voting process. By utilizing Java's collection framework—specifically HashSet, ArrayList, and HashMap—the system ensures reliable handling of voter registration, candidate listing, and vote counting. The use of HashSet effectively eliminates duplicate voter entries and prevents multiple voting attempts, thereby enhancing the integrity and fairness of the election process. The ArrayList provides an organized structure for maintaining candidate information, while the HashMap offers an efficient mechanism for tracking and updating vote counts in real time.

Throughout the implementation, the system successfully guided users through the core stages of the election cycle, including candidate registration, voter registration, voting execution, and final result generation. Each module operated cohesively to deliver a seamless and error-free voting experience. The final outcome accurately identified the candidate with the highest votes, confirming the correctness and reliability of the program's logic.

Overall, this project highlights how fundamental data structures and object-oriented programming concepts can be applied to address practical problems in election management. The system serves as a foundational model that can be expanded with advanced features such as data persistence, secure authentication, graphical interfaces, and enhanced privacy mechanisms. This implementation validates the effectiveness of Java as a tool for building secure, organized, and scalable voting solutions.

REFERENCES

[1] Oracle Java Documentation – Collections Framework Overview. Oracle Corporation. https://docs.oracle.com/javase/8/docs/technotes/guides/collections/

[2] Schildt, H. (2018). Java: The Complete Reference (11th Edition). McGraw-Hill Education.

[3] Horstmann, C. S., & Cornell, G. (2015). Core Java Volume I – Fundamentals (10th Edition). Pearson Education.

[4] Savitch, W. J. (2014). Absolute Java (5th Edition). Addison-Wesley.

[5] Deitel, P. J., & Deitel, H. M. (2017). Java: How to Program (Early Objects Version). Pearson.

[6] TutorialsPoint. Java Collections Framework — HashSet, ArrayList, HashMap. https://www.tutorialspoint.com/java/java_collections.htm

[7] GeeksforGeeks. Java Collection Framework – Introduction and Examples. https://www.geeksforgeeks.org/collections-in-java/

[8]Dr.J.NarendraBabu, upcoming Strengths on Internet of Things Journal of Advanced Research in Dynamical & Control systems Vol.11,N0.11,2019

[9]Dr.J.Naredra Babu, Enhancement of RVM Using FPGA Journal of Advanced Research in Dynamical & Control systems Vol.11,No.11,2019

[10]Dr.J.NarendrBabu, BrainSignal processing : Analysis, Technologies and Application Journal of Advanced Research in Dynamical & Control systems (JARDCS)Vol.11,No: 12,2019.

[11] Dr.J.NarendraBabu, Steganography based message hiding in image, International journal of advance research in science and engineering, vol.no.4, Issue No.12, December 2015, ISSN-2319-8354

[12] Dr.J.Narendra Babu, et.al- Experimental detection of vehicles for toll gate billing, International journal of Science technology and management, vol.no.4, Issue No.12, December 2015, ISSN 2394-1537

Author Biography

Dr. J.Narendra Babu is a seasoned academician with over 28 years of experience in teaching and software industry, Dr.J.Narendra Babu currently serves as a Professor in the Department of Data Science at Sapthagiri NPS University. He holds a B.Tech, M.Tech and PhD degree. Dr.J.Narendra Babu has published extensively in reputed journals and conferences, Dr.J.Narendra Babu has played a key role in With mentoring students, supervising undergraduate projects, coordinating academic activities, and contributing to curriculum development