```
import hashlib
import uuid
from cryptography.fernet import Fernet
import os

# Encryption key generation
key = Fernet.generate_key()
cipher_suite = Fernet(key)

# Simulated database using dictionaries
voters_db = {}
candidates_db = {}
votes_db = {}
audit_log = []

class Voter:
    def __init__(self, voter_id):
        self.voter_id = voter_id
        self.has_voted = False

class VotingSystem:
    def __init__(self):
        self.voters = voters_db
        self.candidates = candidates_db
        self.votes = votes_db
        self.audit_log = audit_log

    def register_voter(self, voter_id):
        if voter_id not in self.voters:
            self.voters[voter_id] = Voter(voter_id)
            print(f"Voter {voter_id} registered successfully.")
            self.log_action(f"Registered voter {voter_id}.")
        else:
            print("Voter ID already registered.")

    def add_candidate(self, candidate_name):
        if candidate_name not in self.candidates:
            self.candidates[candidate_name] = 0
            print(f"Candidate {candidate_name} added successfully.")
            self.log_action(f"Added candidate {candidate_name}.")
        else:
            print("Candidate already exists.")

    def cast_vote(self, voter_id, candidate_name):
        if voter_id in self.voters and not self.voters[voter_id].has_voted:
            if candidate_name in self.candidates:
                vote_id = uuid.uuid4().hex
                vote_hash = hashlib.sha256(vote_id.encode()).hexdigest()

                # Encrypting vote information
                encrypted_vote = cipher_suite.encrypt(candidate_name.encode())

                # Recording the vote
                self.votes[vote_hash] = encrypted_vote
                self.candidates[candidate_name] += 1
                self.voters[voter_id].has_voted = True
                print(f"Vote cast successfully by voter {voter_id}.")
                self.log_action(f"Vote cast by {voter_id} for {candidate_name}.")
            else:
                print("Candidate does not exist.")
        else:
            print("Invalid voter ID or vote already cast.")

    def display_results(self):
        print("\n--- Election Results ---")
        for candidate, votes in self.candidates.items():
            print(f"{candidate}: {votes} votes")
        self.log_action("Displayed election results.")

    def verify_votes(self):
        print("\n--- Verifying Votes ---")
        for vote_hash, encrypted_vote in self.votes.items():
            candidate_name = cipher_suite.decrypt(encrypted_vote).decode()
            print(f"Vote ID: {vote_hash} verified for candidate {candidate_name}.")
        self.log_action("Verified all votes.")

    def log_action(self, action):
        log_entry = f"{action} - {uuid.uuid4().hex}"
        self.audit_log.append(log_entry)
        print(f"Action logged: {log_entry}")

    def view_audit_log(self):
        print("\n--- Audit Log ---")
        for entry in self.audit_log:
            print(entry)

def main():
    system = VotingSystem()
    print("Welcome to the Advanced Electronic Voting System")

    while True:
        print("\nOptions:")
        print("1. Register Voter")
        print("2. Add Candidate")
        print("3. Cast Vote")
        print("4. Display Results")
        print("5. Verify Votes")
        print("6. View Audit Log")
```

```
        print("7. Exit")

        choice = input("\nEnter your choice (1-7): ")

        if choice == '1':
            voter_id = input("Enter voter ID: ")
            system.register_voter(voter_id)

        elif choice == '2':
            candidate_name = input("Enter candidate name: ")
            system.add_candidate(candidate_name)

        elif choice == '3':
            voter_id = input("Enter voter ID: ")
            candidate_name = input("Enter candidate name: ")
            system.cast_vote(voter_id, candidate_name)

        elif choice == '4':
            system.display_results()

        elif choice == '5':
            system.verify_votes()

        elif choice == '6':
            system.view_audit_log()

        elif choice == '7':
            print("\nExiting the system. Goodbye!")
            break

        else:
            print("\nInvalid choice! Please select a valid option.")

if __name__ == "__main__":
    main()
```

```
5. Verify Votes
6. View Audit Log
7. Exit

Enter your choice (1-7): 4

--- Election Results ---
Avinash: 1 votes
Lokesh: 0 votes
Action logged: Displayed election results. - 68ce31f7e064428e8374c3879cfed2b1

Options:
1. Register Voter
2. Add Candidate
3. Cast Vote
4. Display Results
5. Verify Votes
6. View Audit Log
7. Exit

Enter your choice (1-7): 5

--- Verifying Votes ---
Vote ID: 762b42badfa24d4d7aaace394d71140f905d70c591a56d04145dcfb8a44dad63 verified for candidate Avinash.
Action logged: Verified all votes. - 20e3700cd61c43d6b292be57be381235

Options:
1. Register Voter
2. Add Candidate
3. Cast Vote
4. Display Results
5. Verify Votes
6. View Audit Log
7. Exit

Enter your choice (1-7): 6

--- Audit Log ---
Registered voter DSA1554. - 16fc1565f2da45e98bca7e130c5f3867
Added candidate Avinash. - ec94c50b3fe64703b282117c009d2acf
Registered voter ASD0088. - 2bac7f1f0b574358a45bd45724260681
Added candidate Lokesh. - ae0f466942424361ad1d22602d6077dc
Vote cast by DSA1554 for Avinash. - b93f43dd453542c5acbf38a3cdc7ea92
Displayed election results. - 68ce31f7e064428e8374c3879cfed2b1
Verified all votes. - 20e3700cd61c43d6b292be57be381235

Options:
1. Register Voter
2. Add Candidate
3. Cast Vote
4. Display Results
5. Verify Votes
6. View Audit Log
7. Exit

Enter your choice (1-7): 7

Exiting the system. Goodbye!
```