EXERCISE NUMBER: 8 **REGISTER NUMBER:RA2111026050026**

DATE	
SUBMITTED BY	**VISHNUPRIYAN S**
TITLE / ROLE	**Online Voting System**

PLANNING PHASE:

Planning for an online voting system requires careful consideration of various factors, including security, accessibility, usability, scalability, and legal compliance. Here are some key steps to follow:

Identify the stakeholders: Determine who will be using the online voting system, including voters, election officials, and other relevant parties.

Define the requirements: Identify the functional and non-functional requirements of the online voting system, such as the ability to register voters, verify their identity, provide ballot choices, and securely tally and report results.

Evaluate technology options: Research available technologies, including proprietary and open-source software, and consider the pros and cons of each. Evaluate the infrastructure requirements, including hardware, software, and network connectivity.

Define the security measures: Define the security measures that will be implemented to protect the online voting system from tampering, fraud, and cyberattacks. Consider using encryption, multi-factor authentication, and other security measures to ensure the integrity of the voting process.

Design the user interface: Design a user-friendly interface that is easy to navigate and accessible to all users, including those with disabilities.

Develop a testing plan: Develop a testing plan to ensure that the online voting system is functioning as intended and meets the requirements defined in step two.

Develop a deployment plan: Develop a deployment plan that outlines the steps required to deploy the online voting system, including training election officials and voters on how to use the system.

Ensure legal compliance: Ensure that the online voting system complies with all relevant laws and regulations, including those related to privacy, data protection, and election integrity.

By following these steps, you can help ensure that your online voting system is secure, accessible, and compliant with legal requirements.

PYTHON PROGRAM FOR PLANNING PHASE IN ONLINE VOTING SYSTEM

```
# Import necessary libraries
import datetime
import json

# Define the election start and end times
start_time = datetime.datetime(2023, 4, 10, 9, 0, 0)
end_time = datetime.datetime(2023, 4, 15, 17, 0, 0)

# Define the candidates for the election
candidates = ['Candidate A', 'Candidate B', 'Candidate C']

# Define the number of allowed votes per voter
allowed_votes = 1

# Define the path for storing the election data
data_path = 'election_data.json'

# Define a function for saving the election data to a file
def save_election_data(data):
    with open(data_path, 'w') as f:
        json.dump(data, f)

# Define the initial election data
election_data = {
    'start_time': start_time.isoformat(),
    'end_time': end_time.isoformat(),
    'candidates': candidates,
    'allowed_votes': allowed_votes,
    'votes': {}
}

# Save the election data to a file
save_election_data(election_data)
```

DESIGN PHASE:

The design phase for an online voting system involves translating the requirements identified in the planning phase into a concrete system design. Here are some key steps to follow:

System architecture design: Determine the overall system architecture, including the hardware, software, and network components. Consider the scalability requirements and plan for future growth and expansion.

User interface design: Design the user interface to be intuitive, user-friendly, and accessible to all voters, including those with disabilities. Ensure that the interface is easy to navigate, and that the instructions are clear and concise.

Data modeling: Develop a data model that defines the data structures and relationships required to support the online voting system. This includes defining the data entities, attributes, and relationships required to support the system.

Security design: Develop a security design that ensures the integrity and confidentiality of the voting process. This includes defining the security measures required to protect the system from external threats, such as hacking or unauthorized access.

Verification and validation design: Develop a verification and validation plan that ensures the system is meeting the requirements defined in the planning phase. This includes defining the testing methodologies, test cases, and acceptance criteria.

System integration design: Define the system integration requirements, including the interfaces and protocols required to integrate with external systems, such as voter registration databases or identity verification systems.

Performance design: Define the performance requirements and design the system to ensure that it can handle the expected volume of traffic during peak voting periods.

Disaster recovery and business continuity design: Develop a disaster recovery and business continuity plan that ensures the system can recover from a disaster or outage and continue to function without interruption.

By following these steps, you can help ensure that the online voting system is designed to meet the requirements defined in the planning phase and is secure, accessible, and compliant with legal requirements.

PYTHON PROGRAM FOR DESIGN PHASE IN ONLINE VOTING SYSTEM

```
# Import necessary libraries
import datetime
import json
import random

# Define the path for the election data file
data_path = 'election_data.json'

# Define a function for loading the election data from the file
def load_election_data():
    with open(data_path, 'r') as f:
        data = json.load(f)
    return data

# Load the election data
election_data = load_election_data()

# Define a function for generating a random voter ID
def generate_voter_id():
    voter_id = ''
    for i in range(6):
        voter_id += str(random.randint(0, 9))
    return voter_id

# Define a function for submitting a vote
def submit_vote(voter_id, candidate):
    if voter_id in election_data['votes']:
        if len(election_data['votes'][voter_id]) < election_data['allowed_votes']:
            election_data['votes'][voter_id].append(candidate)
        else:
            return False
    else:
        election_data['votes'][voter_id] = [candidate]
    save_election_data(election_data)
    return True

# Define a function for getting the current time
def get_current_time():
    return datetime.datetime.now()

# Define a function for checking if the election is open
def is_election_open():
    current_time = get_current_time()
    start_time = datetime.datetime.fromisoformat(election_data['start_time'])
    end_time = datetime.datetime.fromisoformat(election_data['end_time'])
    if current_time >= start_time and current_time <= end_time:
        return True
    else:
        return False

# Define a function for checking if a voter is eligible to vote
def is_eligible_to_vote(voter_id):
    if voter_id in election_data['votes']:
        if len(election_data['votes'][voter_id]) < election_data['allowed_votes']:
            return True
        else:
            return False
    else:
        return True

# Define a function for saving the election data to a file
def save_election_data(data):
    with open(data_path, 'w') as f:
        json.dump(data, f)

# Example usage:
if is_election_open():
    voter_id = generate_voter_id()
    if is_eligible_to_vote(voter_id):
        candidate = random.choice(election_data['candidates'])
        submit_vote(voter_id, candidate)
        print(f"Vote submitted for {candidate} by voter {voter_id}.")
    else:
        print("This voter is not eligible to vote.")
else:
    print("The election is not currently open.")
```

DEVELOPMENT PHASE:

The development phase for an online voting system involves implementing the system design created in the design phase. Here are some key steps to follow:

Software development: Develop the software components of the online voting system, including the front-end interface, back-end server, and database.

Security implementation: Implement the security measures defined in the design phase, including encryption, multi-factor authentication, and other security measures to protect against tampering, fraud, and cyberattacks.

Testing and quality assurance: Conduct testing and quality assurance activities to ensure that the online voting system is functioning as intended and meets the requirements defined in the planning and design phases.

System integration: Integrate the online voting system with any external systems or databases required, such as voter registration databases or identity verification systems.

Performance optimization: Optimize the system performance to ensure that it can handle the expected volume of traffic during peak voting periods and meet the performance requirements defined in the design phase.

Accessibility testing: Conduct accessibility testing to ensure that the online voting system is accessible to all voters, including those with disabilities.

User acceptance testing: Conduct user acceptance testing to ensure that the online voting system is easy to use, intuitive, and meets the needs of voters.

Deployment: Deploy the online voting system to the production environment, following the deployment plan created in the planning phase.

By following these steps, you can ensure that the online voting system is developed to meet the requirements defined in the planning and design phases, is secure, accessible, and compliant with legal requirements, and is ready to be used by voters during an election.

PYTHON PROGRAM FOR DEVELOPMENT PHASE IN ONLINE VOTING SYSTEM

```
# Import necessary libraries
import datetime
import json
import hashlib
from flask import Flask, jsonify, request

# Define the path for the election data file
data_path = 'election_data.json'

# Define a function for loading the election data from the file
def load_election_data():
    with open(data_path, 'r') as f:
        data = json.load(f)
    return data

# Load the election data
election_data = load_election_data()

# Define a Flask app
app = Flask(_name_)

# Define a route for submitting a vote
@app.route('/submit_vote', methods=['POST'])
def submit_vote():
    data = request.json
    voter_id = data['voter_id']
    candidate = data['candidate']
    if is_election_open() and is_eligible_to_vote(voter_id):
        election_data['votes'][voter_id] = hash_vote(candidate)
        save_election_data(election_data)
        return jsonify({'status': 'success'})
    else:
        return jsonify({'status': 'error', 'message': 'Vote not submitted.'})

# Define a function for hashing a vote
def hash_vote(candidate):
    vote_string = candidate + election_data['hash_key']
    return hashlib.sha256(vote_string.encode()).hexdigest()

# Define a function for getting the current time
def get_current_time():
    return datetime.datetime.now()

# Define a function for checking if the election is open
def is_election_open():
    current_time = get_current_time()
    start_time = datetime.datetime.fromisoformat(election_data['start_time'])
    end_time = datetime.datetime.fromisoformat(election_data['end_time'])
    if current_time >= start_time and current_time <= end_time:
        return True
    else:
        return False
```

```
# Define a function for checking if a voter is eligible to vote
def is_eligible_to_vote(voter_id):
    if voter_id in election_data['votes']:
        return False
    else:
        return True

# Define a function for saving the election data to a file
def save_election_data(data):
    with open(data_path, 'w') as f:
        json.dump(data, f)

# Example usage:
if _name_ == '_main_':
    app.run()
```