e-Voting System Prototype for Election Commission of Madhya Pradesh

Problem Statement:

The Election Commission of India (ECI) has tasked you with developing an **online voting system prototype** for the state of Madhya Pradesh. The system must cater to both **administrators** and **voters** with separate portals, each part of the same Python code. Your objective is to create a system that ensures secure and transparent elections, where voters can cast their votes online during a specified campaign period.

Project Requirements:

1. Admin Portal:

The **Admin Portal** is designed for election administrators to manage election campaigns and candidate details. The key functionalities are:

- **Campaign Creation:**
  The admin can create multiple election campaigns (e.g., for MLA elections, local body elections) based on constituencies. Each campaign includes:
  - **Campaign Name**
  - **Campaign ID**
  - **Constituency Name**
  - **Start Date and Time**
  - **End Date and Time**

- **Candidate Management:**
  Under each campaign, the admin can add multiple candidates. For each candidate, the following details will be recorded:
  1. **Candidate ID**
  2. **Candidate Name**
  3. **Party Symbol** (e.g., Lotus, Hand, etc.)
  4. **Party Name**
  5. **Additional Information** (optional)

- **Campaign Scheduling:**
  The admin must specify the start and end date/time for each campaign. Voters can only cast their vote during the active period of a campaign.
- **Results Publication:**
  After a campaign ends, the admin can publish the results.
- **Login Functionality:**
  The admin logs in using predefined credentials. After three unsuccessful login attempts, access will be locked for security purposes.

2. Voter Portal:

The **Voter Portal** is designed for registered voters to securely log in and cast their votes during an active campaign. The key functionalities are:

- **Voter Authentication:**
  Voters must log in using their **Voter ID** and **Password**, which will be stored in a pre-defined Excel file (dataset). This file will have the following fields:
  - **Voter ID**
  - **Name**
  - **Password**

- **Campaign Participation:**
  - Voters can only view and participate in active campaigns (campaigns that have started but not yet ended).
  - Each voter can cast their vote **once per campaign**.

- **Login Security:**
  If a voter enters incorrect credentials more than three times, they will be locked out of the system.

Special Considerations:

1. **Login Attempts Restriction:**
   Both admin and voter users are allowed **only three incorrect login attempts** before access is blocked for security.
2. **Pre-defined Credentials:**
   - Use predefined admin credentials directly in the Python code.
   - The voter credentials (Voter ID and Password) will be stored in an Excel file (e.g., voter_data.xlsx).
3. **Data Management:**
   The system should be designed to handle voter and candidate data efficiently. Use external Excel files or Python databases (like SQLite) to store:
   - Voter login details
   - Election campaign data
   - Candidate details
   - Voting records

Database and Structure:

- **Voter Dataset (Excel):**
  - Columns: VoterID, Name, Password
- **Campaign Data:**
  - Campaigns are managed as Python objects or stored in a database/excel file with fields: CampaignID, Constituency, StartDate, EndDate
- **Candidate Data:**
  - Candidate details per campaign are stored with fields: CampaignID, CandidateID, CandidateName, PartySymbol, PartyName

- **Vote Records:**
  - Voter participation can be stored in a table with fields: CampaignID, VoterID, Voted (Yes/No)

Functional Flow:

1. **Admin Portal Flow:**
   - Admin logs in using predefined credentials.
   - Admin creates a campaign by providing required details and adding candidates.
   - Admin can view all campaigns, manage candidate details, and publish results.
   - Admin can log out after use.
2. **Voter Portal Flow:**
   - Voter logs in using credentials from the pre-defined Excel file.
   - After successful login, the voter can view live campaigns.
   - The voter casts their vote and confirms their choice.
   - The vote is recorded, and the voter cannot vote again in the same campaign.
   - Voter logs out after voting.

Deliverables:

- **Python Code:**
  - Code for Admin and Voter portals with login functionality.
  - Code for campaign management, candidate addition, and voting mechanism.
  - Handling of Excel files for voter data and candidate/campaign details.
- **Excel Files:**
  - Pre-defined voter dataset.
  - Storage and manipulation of campaign and candidate data.

Tools & Libraries:

- **Python 3.x**
- **Pandas** for handling Excel files.

Optional Tools

- **SQLite** (optional) for database management.
- **Flask** or other Python web frameworks (optional for web-based interface).

==========