Online polling System

Voting in India is a Constitutional right if one is a citizen over 18 years of age. However, that also makes it optional. It has been a tendency among voters, especially in the urban areas, to treat the voting day as a day of rest. Many citizens feel so lazy to stand in a queue for voting. There can also be citizen whose name will be in their native place. They might feel uncomfortable to travel for voting. These problems of voting at the polls can be avoided if there is a facility to vote online.

Creating an Online Polling System Project to enhance the voting process in India requires several key components and functionalities. Here's a more detailed explanation of each requirement:

1. **Voter Registration**:
    - The system should allow the polling administrator to register eligible citizens as voters. During this registration process, the admin should verify that the citizen is over 18 years of age, as this is the constitutional requirement for voting.
2. **Nominee Registration**:
    - The system should provide a platform for the administrator to register candidates (nominees) who are contesting in the election. The nominee details, including their name, address, symbol, political party or constitution, and the district they are running in, should be added to the system.
3. **Voting Process**:
    - Citizens should be able to participate in the election by logging into the system using their unique login ID and password. Once logged in, they should have access to the details of the nominees in their district. Citizens can cast their votes on the designated voting day, ensuring that the process remains secure and confidential.
4. **Result Publication**:
    - After the completion of polling at all locations, the system should be capable of immediately publishing the election results. This transparency is essential to maintain the integrity of the electoral process.
5. **Report Generation**:
    - The system should offer various reporting functionalities to provide insights and transparency into the election process.

    5.a. **Percentage of Voting in Each District**:
        - The system should generate reports showing the percentage of voter turnout in each district. This information can be crucial for analyzing voter engagement and participation.

    5.b. **Party-Wise Voting Count**:
        - The system should also produce reports that break down the voting count by political parties. This data can be used to determine the popularity of specific parties or candidates.

    5.c. **District-Wise Nominee List**:
        - Generating a list of nominees by district can help citizens understand their options and make informed voting decisions.

Report generation

1. To view the percentage of voting in each district.
2. Party wise voting count
3. District wise nominee list

Required Software

- **Eclipse IDE:** An integrated development environment for software development.
- **Xampp/MySQL:** A software stack for local web development, including the MySQL database.

Required Jar files

MySQL connector based on the MySQL database version.

Modules Required

The application should consist of 4 modules

a Voter Module
b Nominee Module
c Vote Module
d Result Module

Tables required

Voter

Column Name	Data Type
VOTER_ID	VARCHAR(25) **primary_key**
VOTER_NAME	VARCHAR(25)
DOB	DATE
AGE	INT
LOGIN_ID	VARCHAR(25)
PASSWORD	VARCHAR(25)
ADDRESS	VARCHAR(50)
DISTRICT	VARCHAR(25)
MOBILE_NUMBER	BIGINT(20)

Nominee

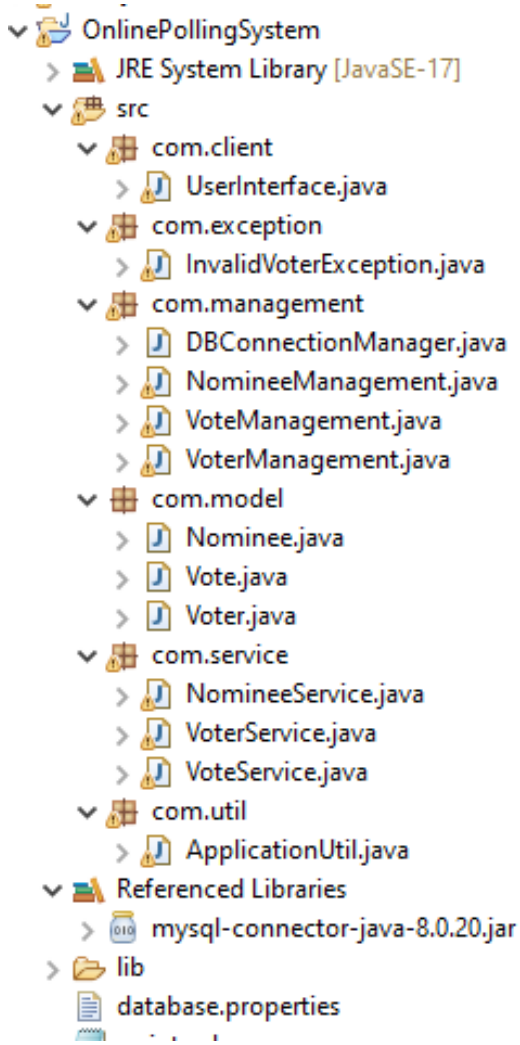
Column Name	Data Type
NOMINEE_ID	VARCHAR(25) **primary_key**
NOMINEE_NAME	VARCHAR(25)
CONSTITUTION	VARCHAR(30)

Column Name	Data Type
DISTRICT	VARCHAR(50)
SYMBOL	VARCHAR(50)
ADDRESS	VARCHAR(100)
VOTE_COUNT	INT

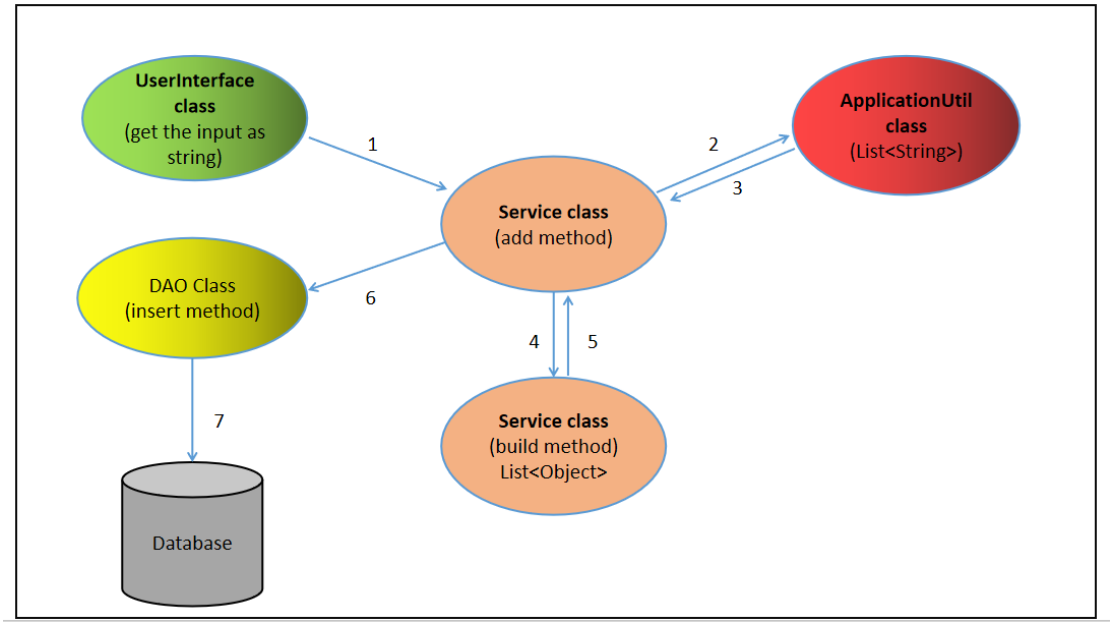
Vote

Column Name	Data Type
vote_id	VARCHAR(20) **primary_key**
voter_id	VARCHAR(20) **foreign_key**
nominee_id	VARCHAR(20) **foreign_key**
voted_date	DATE

Project Structure



Project Flow:



Scope of the Service Module:

Module	Responsibility
NomineeService	The responsibility of the "NomineeService" module is to provide a set of functions and methods for managing nominee data in the context of an election system. This module acts as an intermediary between the application logic and the underlying data storage (typically a database). Its primary responsibilities include: **Building Nominee Lists:** Parsing and processing nominee records to create a list of Nominee objects with attributes like ID, name, constitution, district, symbol, and address. **Adding Nominee Records:** Allowing the addition of nominee records to the system. It takes nominee information as input, converts it into Nominee objects, and stores them in the database through the "NomineeManagement" class. **Modifying Nominee Address:** Enabling the update of a nominee's address in the database by providing the nominee's ID and the new address. **Retrieving Nominee Information:** Retrieving the current address of a nominee based on their unique nominee ID. If the nominee is found in the database, their address is returned. **Retrieving Nominee Records by District:** Retrieving nominee records specific to a given district. It formats the data for display, including column names, a separator line, and the nominee records themselves. **Determining Winning Constitution:**

	Identifying the constitution or constituency with the winning nominee. This information is typically determined based on voting results and stored in the database. **Collecting Party-Wise Voting Counts:** Gathering voting counts for different political parties and organizing them into a map. The map associates party names with their respective vote counts. **Deleting Nominee Records:** Allowing the removal of a nominee record from the database based on the nominee's ID.
VoterService	The "**VoterService**" module is responsible for managing voter-related operations within an election system. Its primary responsibilities include: **Building Voter Lists**: Parsing and processing voter records to create a list of **Voter** objects, including attributes like ID, name, date of birth, age, login ID, password, address, district, and mobile number. **Adding Voter Records**: Enabling the addition of voter records to the system. It takes voter information as input, converts it into **Voter** objects, and stores them in the database through the "**VoterManagement**" class. **Generating Voter IDs**: Creating unique voter IDs by generating sequential numbers and appending them to a prefix. This ensures that each voter has a distinct ID. **Modifying Voter Phone Number**: Allowing the update of a voter's phone number in the database based on the voter's ID. **Retrieving Current Phone Number**: Retrieving the current phone number of a voter based on their voter ID. If the voter is found in the database, their phone number is returned; otherwise, a "Voter not found" message is returned. **Modifying Voter Address**: Enabling the update of a voter's address in the database based on the voter's ID. **Retrieving Current Address**: Retrieving the current address of a voter based on their voter ID. If the voter is found, their address is returned; otherwise, a "Voter not found" message is returned. **Retrieving Voter Records Based on Voter ID**: Fetching voter records based on a specific voter ID and formatting the data for display. **Retrieving Voter Records Based on District**: Retrieving voter records specific to a given district and formatting the data for presentation, including the voter's ID, name, date of birth, age, login ID, password, address, district, and mobile number. **Deleting Voter Records**:

	Allowing the removal of a voter record from the database based on the voter's ID.
VoteService	The "**VoteService**" module is responsible for managing voting-related operations within an election system. Its key responsibilities include: **Building Vote Lists**: Parsing and processing vote records to create a list of **Vote** objects, including attributes such as vote ID, voter ID, nominee ID, voted date, and other relevant data. It generates a unique vote ID for each vote record. **Adding Vote Records**: Facilitating the addition of vote records to the system. It processes vote information, associates it with the corresponding voter and nominee, and records the vote in the database through the "VoteManagement" class. Additionally, it updates the vote count for the nominated candidate. **Generating Vote IDs**: Creating unique vote IDs by generating sequential numbers and appending them to a prefix. This ensures that each vote has a distinct ID. **Calculating Voting Percentage by District**: Retrieving and calculating the voting percentage by district, providing insights into voter participation in different geographical areas. This information is gathered from the "VoteManagement" class. **Retrieving District of Voter**: Determining the district of a voter based on their login ID and password. It retrieves the voter's district from the database through the "VoteManagement" class. **Deleting Enrollment Records**: Allowing the removal of enrollment records based on their enrollment ID. This operation is performed through the "VoteManagement" class.

Scope of the Management Module:

Module	Responsibility
NomineeManagement	The "**NomineeManagement**" module is responsible for handling various database operations related to nominee data within an election system. Its primary responsibilities include: **Inserting Nominee List:** Inserting a list of nominee records into the database. This operation is performed using batch insertion, which allows multiple nominee records to be inserted efficiently. **Updating Nominee Address:**

	Updating the address of a nominee in the database based on the nominee's ID. This operation is used to keep nominee information up to date. **Retrieving Nominee List Based on Nominee ID:** Retrieving a list of nominees based on their unique nominee ID. This is useful for looking up specific nominees by their ID. **Retrieving Nominee List by District:** Retrieving a list of nominees specific to a given district. This operation allows the retrieval of nominee records based on a geographical criterion. **Declaring Winning Constitution:** Determining the winning constitution or constituency based on the total vote counts for each constitution. The winning constitution is the one with the highest vote count. **Retrieving Party-Wise Vote Counts:** Collecting and providing a map that associates each constitution or party with its total vote count. This information helps in analyzing the election results. **Deleting Nominee Records:** Allowing the removal of nominee records from the database based on the nominee's ID. This operation facilitates the management of nominee data.
VoterManagement	The "**VoterManagement**" module is responsible for handling various database operations related to voter data within an election system. Its primary responsibilities include: **Inserting Voter List:** Inserting a list of voter records into the database. This operation is performed using batch insertion, allowing multiple voter records to be inserted efficiently. **Updating Voter Details:** Updating details of a voter in the database, including their phone number and address. These operations are used to keep voter information up to date. **Checking if Voter ID Exists:**

	Verifying whether a given voter ID already exists in the database. This operation is important to avoid duplicate voter records. **Retrieving Voter List Based on Voter ID:** Retrieving a list of voters based on their unique voter ID. This is useful for looking up specific voters by their ID. **Retrieving Voter List by District:** Retrieving a list of voters specific to a given district. This operation allows the retrieval of voter records based on a geographical criterion. **Deleting Voter Records:** Allowing the removal of voter records from the database based on the voter's ID. This operation facilitates the management of voter data.
VoteManagement	The "**VoteManagement**" module is responsible for managing various database operations related to voting data within an election system. Its primary responsibilities include: **Inserting Vote List**: Inserting a list of vote records into the database. This operation is performed using batch insertion, allowing multiple vote records to be inserted efficiently. **Updating Nominee Vote Count**: Updating the vote count of a nominee in the database. This operation is used to increment the vote count of a nominee when a vote is cast for them. **Updating Enrollment Fee Status**: Updating the fee status of an enrollment record in the database. This operation is used to manage the payment status of enrolled voters. **Retrieving Voter Details**: Retrieving voter details based on their login ID and password. This operation is crucial for authenticating voters during the voting process.

	Checking if Vote ID Exists: Verifying whether a given vote ID already exists in the database. This is important to prevent duplicate vote records. **Calculating Voting Percentage by District**: Calculating the voting percentage for each district based on the number of votes cast and the total number of registered voters. This operation provides insights into voter turnout. **Getting the Number of Voters in a District**: Determining the number of voters in a specific district who have cast their votes. **Getting the Total Registered Voters in a District**: Obtaining the total number of registered voters in a district, which is used to calculate voting percentages. **Deleting Enrollment Records**: Allowing the removal of enrollment records from the database based on the enrollment ID. This operation is used for enrollment management.

Scope of Database module:

Module	Responsibilities
DBConnectionManager	**The DBConnectionManager module** is responsible for managing and providing database connections to other modules in the application. Its primary responsibilities include: **Loading Database Properties:** It reads the database connection properties from a configuration file, such as database.properties, which typically contains details like the database URL, driver name, username, and password. **Establishing Database Connection:** It establishes a connection to the database using the properties loaded from the configuration file. This connection is created through the Java Database Connectivity (JDBC) API.

	Handling Exceptions: It handles exceptions related to file IO (e.g., IOException), database drivers (e.g., ClassNotFoundException), and SQL errors (e.g., SQLException) that may occur during the connection process.

Scope of the Exception module

Module Name	Responsibility
InvalidVoterException	The **InvalidVoterException** class is a custom exception class used to handle exceptions related to invalid voter records in the application. It extends the standard Exception class and includes a single-argument constructor to set the exception message. The exception message is passed to the super class constructor.