USECASE VIEW:

IDENTIFICATION OF ACTORS:

- Actors are NOT part of the system.
- Actors represent anyone or anything that interacts with (input to or receives output from) the system.
- An actor is someone or something that:
  - Interacts with or uses the system.
  - Provides input to and receives information from the system
  - Is external to the system and has no control over the use cases.
- Actors are discovered by examining:
  - Who directly uses the system?
  - Who is responsible for maintaining the system?
  - External hardware used by the system.
  - Other systems that need to interact with the system.
- The needs of the actor are used to develop use cases. This insures that the system will be what the user expected.

<u>Graphical Depiction</u>

- An actor is a stereotype of a class and is depicted as a "stickman" on a use-case diagram.
  Naming: The name of the actor is displayed below the icon.

Questions that help to identify actors :

1. Who is interested in a certain requirement?

2. Where is the system used within the organization?

3. Who will benefit from the use of the system?

4. Who will supply the system with information, use this information, and remove this information?

5. Who will support and maintain the system?

6. Does the system use an external resource?

7. Does one person play several different roles?

8. Do several people play the same role?

9. Does the system interact with a legacy system?

Using the above questions we have identified two actors in Online Voting System.They are:
1.Administrator

2. Voter

Admin: Administrator is a person who is responsible le for registration of voter ,preparation of ballot form , updating the number of votes and prepare the results.

Voter: The person who register for voting can vote in the particular period of time.

IDENTIFICATION OF USE-CASES AND SUB USE-CASES

Use case is a sequence of transactions performed by a system that yields a measurable result of values for a particular actor. The use cases are all the ways the system may be used.

Graphical Depiction:

- The basic shape of a use case is an ellipse:

Naming:

- A use case may have a name, although it is typically not a simple name. It is often written as an informal text description of the actors and the sequences of events between objects. Use case names often start with a verb.

- The name of the use case is displayed above

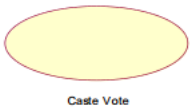
Questions that help to find use cases

1. What are the tasks of each actor?
2. Will any actor create, store, change, remove or read information in the system?
3. What use cases will create, store, change, remove, or read this information?
4. Will any actor need to inform the system about sudden, external changes?
5. Does any actor need to be informed about certain occurrences in the system?
6. What use cases will support or maintain the system?
7. Can all functional requirements be performed by the use cases?

By applying above questions to online voting system application the following use cases are identified .They are

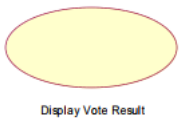
1) **Login**: This use case provides the voter to login into the system.
   UML notation:

2) **Caste Vote:** This use case provides to vote for a particular contestant
   UML notation :



3) **Display Vote Result:** This use case provides to display the vote result of the voter.

   UML notation:



4) **Generate Reports:** This use case is started by admin .It display the winners.

   UML notation:

5) Maintain Candidate information: This use case is started by the admin. It provides to maintain the candidate information.

UML notation:

6)Search the Region: This use case is maintained by the admin to provide the region of the candidate participating.

UML Notation:

FLOW OF EVENTS:

Use case:	**Login**
Id	1
Brief description:	This use case describes how a user logs into the Online Voting System. The actors starting this use case are voter and the administrator of the system
Primary actor	voter, Administrator
Secondary actor	None
Flow of events: Basic Flow	1. The system validates the actor's password and also username and logs him/her into the system. 2. The system displays the Main Form and the use case ends.
Alternate flow:	1.Invalid Name / Password 2.If in the basic flow the system cannot find the name or the password is invalid, an error message is displayed. The actor can type in a new name or password or choose to cancel the operation, at which point the use case ends
Pre condition:	User Should registered
Post condition:	Successfully Log off
Extension Point	Nill

Use case:	**Caste Vote**
Id:	2
Brief description:	The voters can answer to the questions listed on the Internet or voting for a result on the basis of the items listed bellowing list box .

Primary actor	voter, Administrator
Secondary actor	None
Flow of events: Basic Flow	1. Voter login into the voting system . 2. Voter select the thesis that the user want to do the voting . 3. Voter Click the result and finish the voting
Alternate flow:	1.Display Invalid login. 2.Display Invalid candidate selection 3.Display Results not yet declared
Pre condition:	The voter login to the Online voting system so that he or she can do the voting
Post condition:	Voted successfully.
Extension Point	Nill

Use case:	**Maintain candidate information**
Id:	3
Brief description:	This function is used to maintain all of the information about the candidate and the region that he is participating. This function also provides add candidate,delete candidate and update the candidate.
Primary actor	Administrator
Secondary actor	None
Flow of events: Basic Flow	1:Admin login first. 2: Inorder to maintain the candidates information(update,add candidate,delete candidate) like name,address, region of participation and party symbol is maintained.
Alternate flow:	1.Invalid Login 2.Candidate's information option is not available correct time.
Pre condition	1.candidate should be registered by the admin in the specified region. 2.Candidate's information must be collected and maintained in the database.
Post condition:	Candidates information is successfully managed(updated,deleted or added).
Extension Point	Nill

Use case:	**Search the region**
Id:	4
Brief description:	Voter can search the region according to interest
Primary actor	Voter
Secondary actor	None
Flow of events: Basic Flow	1.Voter must login into the system 2.Voter must search the regions according to interest
Alternate flow:	1.Login Failed 2.Region Not found
Pre condition:	Voter must login into the system
Post condition:	Select the election by the voter
Extension Point	Nill

Use case:	**Generate Reports and Result**
Id:	5

Brief description:	The administrator of the system can publish the result of the system
Primary actor	voter, Administrator
Secondary actor	None
Flow of events: Baisc Flow	1. Voter login into the Online Voting System. 2. Voter select the thesis of the online voting System. 3. The voter publish the result of the thesis of the project. 4.The voting result can be seen on the web by clicking the thesis of the system
Alternate flow:	1. Invalid login 2. The login voter doesn't have the authority to publish the Result so that the system provide the voter with an information that he or she is not capable of this function 3. Result not yet declared 4. Message is sent
Pre condition:	1. The administrator login into the Online Voting System. 2. The voter has the authority to publish the result of the voting thesis
Post condition:	The voter can see the result of the online voting system after the result is published.
Extension Point	Nill

Use case:	**Display Vote Result**
Id:	6
Brief description:	The voters can browsing the symbol on the web and see the results of the voting.
Primary actor	voter, Administrator
Secondary actor	None
Flow of events: Baisc Flow	1. Voter login into the Online Voting System. 2. Voter selects the symbol of the online voting System. 3.The voting result can be seen on the web by clicking the thesis of the system
Alternate flow:	1.Invalid Login 2.No symbol found 3.Message is sent
Pre condition:	The voter has login into the Online Voting System
Post condition:	None
Extension Point	Nill

USE-CASE DIAGRAM

Use-case diagrams graphically represent system behavior (use cases). These diagrams present a high level view of how the system is used as viewed from an outsider's (actor's) perspective. A use-case diagram may contain all or some of the use cases of a system.

A use-case diagram can contain:

- Actors ("things" outside the system)
- Use cases (system boundaries identifying what the system should do)
- Interactions or relationships between actors and use cases in the system including the associations, dependencies, and generalizations.

Use-case diagrams can be used during analysis to capture the system requirements and to understand how the system should work. During the design phase, you can use use-case diagrams to specify the behavior of the system as implemented.

RELATIONS:

Association Relationship:

An association provides a pathway for communication. The communication can be between use cases, actors, classes or interfaces. Associations are the most general of all relationships and consequentially the most semantically weak. If two objects are usually considered independently, the relationship is an association

By default, the association tool on the toolbox is uni-directional and drawn on a diagram with a single arrow at one end of the association. The end with the arrow indicates who or what is receiving the communication.

Bi-directional association:

If you prefer, you can also customize the toolbox to include the bi-directional tool to the use- case toolbox.

In An ASSOCIATION Relationship, we can provide Stereotype COMMUNICATE also as shown below

Dependency Relationship:

A dependency is a relationship between two model elements in which a change to one model element will affect the other model element. Use a dependency relationship to connect model elements with the same level of meaning. Typically, on class diagrams, a dependency relationship indicates that the operations of the client invoke operations of the supplier.

We can provide here

1. Include Relationship.

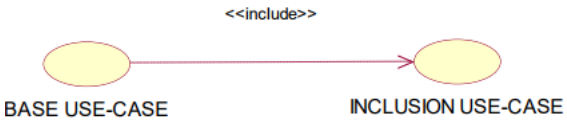
2. Extend Relationship

• There are two types of relationships that may exist between use cases: include relationship and extend relationship.

- Multiple use cases may share pieces of the same functionality. This functionality is placed in a separate use case rather than documenting it in every use case that needs it

- Include relationships are created between the new use case and any other use case that "uses" its functionality.

An include relationship is a stereotyped relationship that connects a base use case to an inclusion use case. An include relationship specifies how behavior in the inclusion use case is used by the base use case.



Extended Relationship:

An extend relationship is a stereotyped relationship that specifies how the functionality of one use case can be inserted into the functionality of another use case. Extend relationships between use cases are modeled as dependencies by using the Extend stereotype.

An extend relationship is used to show

- Optional behavior

- Behavior that is run only under certain conditions such as triggering an alarm

- Several different flows that may be run based on actor selection

- An extend relationship is drawn as a dependency relationship that points from the extension to the base use case

The extend relationship sample demonstrates how you can use an extend relationship to connect use cases. The sample illustrates two important aspects of extend relationships:

· An extend relationship shows optional functionality or system behavior.

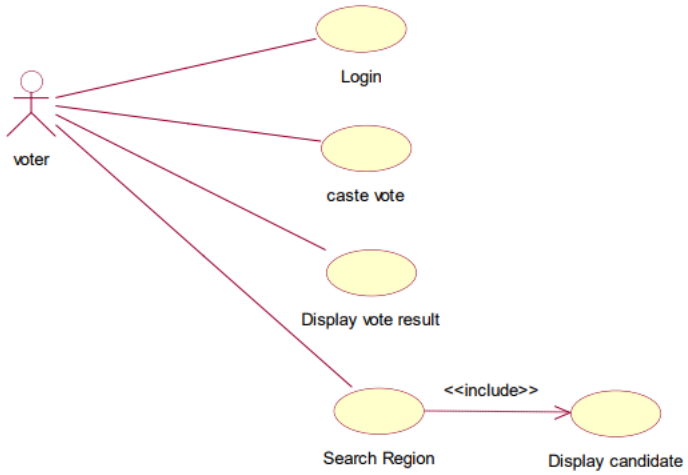
· A base use case does not need to acknowledge any specific extended use cases
Finally we can conclude

«extend» is used when you wish to show that a use case provides additional functionality that may be required in another use case.

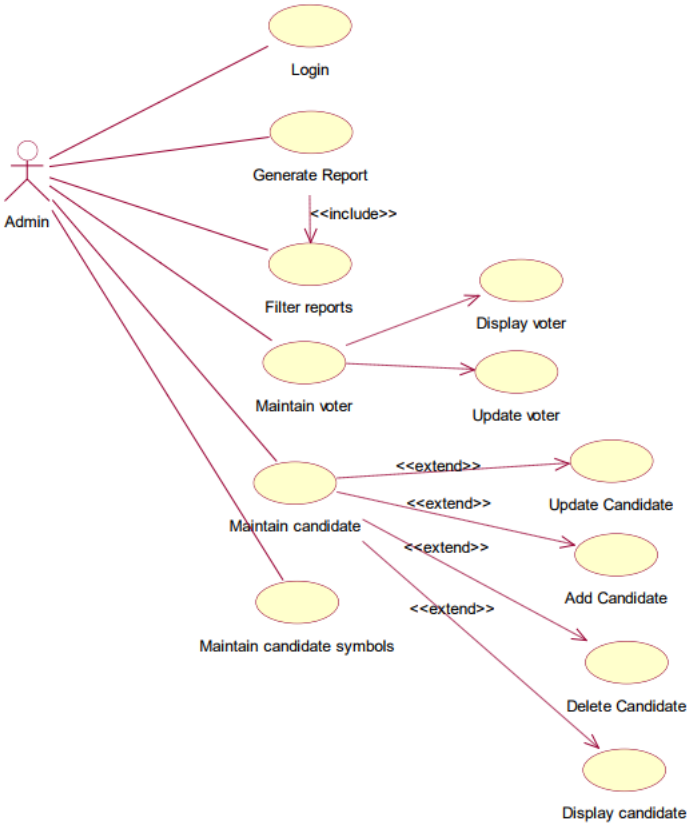
«include» applies when there is a sequence of behavior that is used frequently in a number of use cases, and you want to avoid copying the same description of it into each use case in which it is used.

USECASE DIAGRAM FOR ONLINE VOTING SYSTEM:

USECASE DIAGRAM FOR VOTER



USECASE DIAGRAM FOR ADMINISTRATOR:


Login
Generate Report
Admin
<<include>>
Filter reports
Display voter
Maintain voter
Update voter
<<extend>>
Update Candidate
Maintain candidate
<<extend>>
<<extend>>
Add Candidate
Maintain candidate symbols
<<extend>>
Delete Candidate
Display candidate