

DETAILED DESIGN

Project Code:	EVS-01
Project Name:	Electronic Voting System

Revision History

Version (x.yy)	Date of Revision	Description of Change	Reason for Change	Affected Sections	Approved By
1.0	16/04/2013	Initial Draft			
1.1	30/04/2013	Revision			
2.10	July 2013	Revision			
2.20	Sept-2013	Revision	Mapping with CPC		
2.3	Dec 2013	Revision	Mapping with UCF		

Affected Groups

List of Reference Documents

Name	Version No.
1.RS_EVS	2.20
2.FS_EVS	2.20
3.	
4.	

Prepared by/Date

Reviewed by/Date

Approved by/Date

Table of Contents

TABLE OF CONTENTS	2
1. INTRODUCTION	3
1.1 Background	3
1.2 Purpose	3
1.3 Scope	3
2. GLOBAL DATA STRUCTURES AND SHARED DATA FUNCTIONS	4
3. HIGH LEVEL DESIGN.....	5
3.1 Use Case Diagrams	5
3.1.1 Use Case Diagram for Admin	5
3.1.2 Use Case Diagram for Electoral Officer	6
3.1.3 Use Case Diagram for Voter	6
3.3 Class Diagram	8
3.4 Sequence Diagram	8
3.5 Packages / Classes / Interface	10
3.6 UI Templates	15
4. CRITICAL FUNCTIONS AND FOCUS FOR TESTING	17
5. LIMITATIONS	17
6. APPENDIX	18
1. Table: EVS_TBL_User_Credentials	18
2. Table: EVS_TBL_User_Profile	18
3. Table: EVS_TBL_Election	18
4. Table: EVS_TBL_Party	19
5. Table: EVS_TBL_Candidate	19
6. Table: EVS_TBL_Application	19
7. Table: EVS_TBL_Result	20
8. Table: EVS_TBL_EO	20
9. Table: EVS_TBL_Voter_Details	20
Database Sequences.....	20

1. Introduction

1.1 Background

XYZ Automations Ltd focuses on automating various systems that have been working manually since years

1.2 Purpose

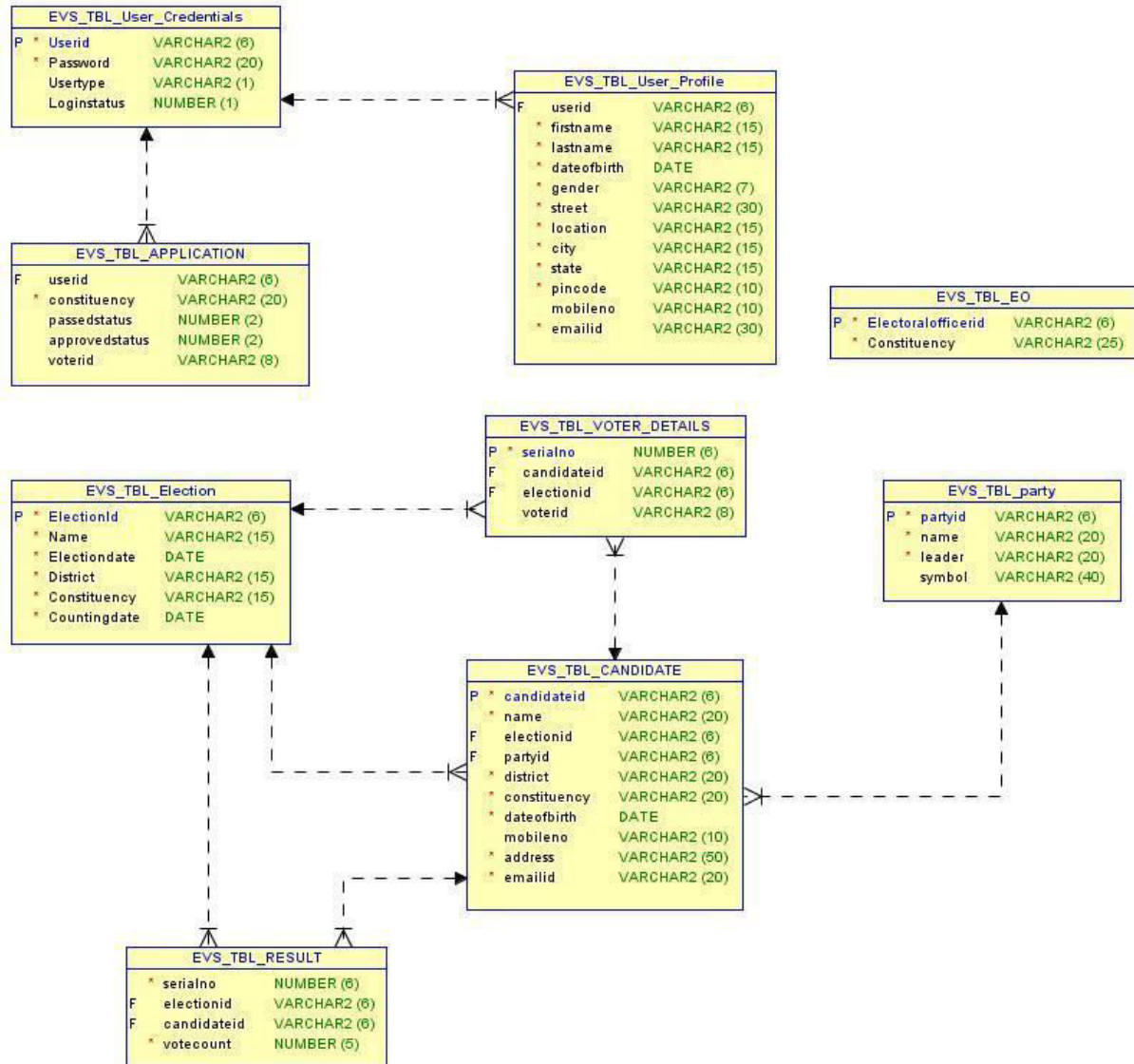
XYZ Automations Ltd has recently planned to develop “Electronic Voting System” - an application to automate the process of planning and managing the voting system, as well as the actual voting activity.

1.3 Scope

The scope of the Electronic Voting System (EVS) will be to provide the functionality as described in Functional Requirements document. The system will be developed on a Windows XP/Windows 7 machine using Java technology and Oracle database.

2. Global Data Structures and Shared Data Functions

This section describes the structure of 9 tables to be used for the implementation of requirements as stated in the specification.



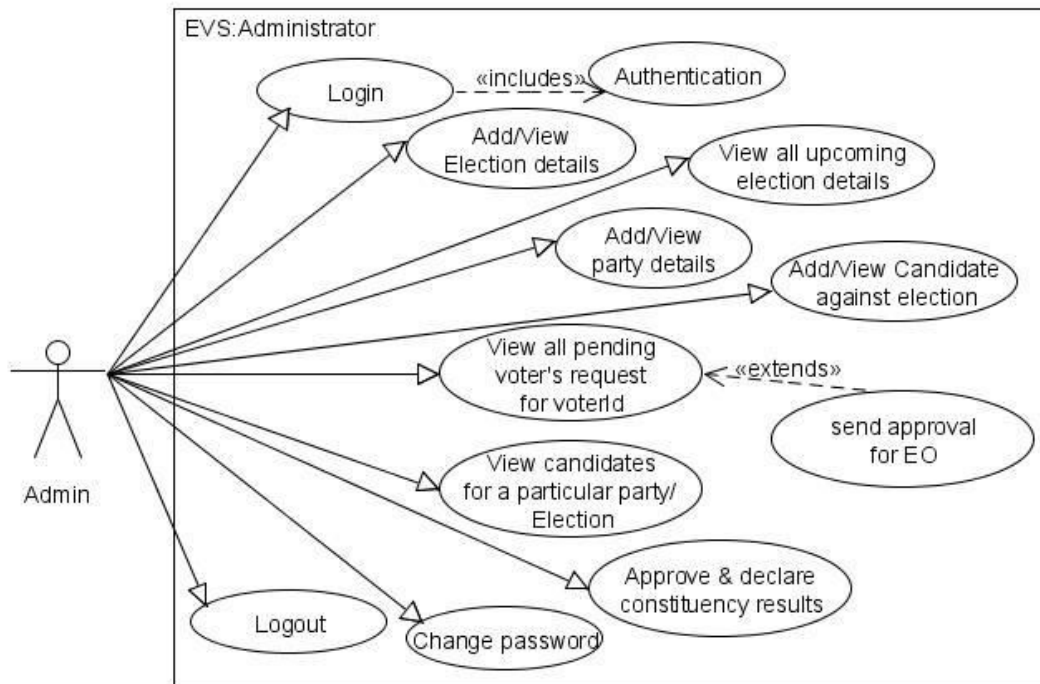
3. High Level Design

This section describes the high level design diagrams. Use case diagram with Use Case definition, Sequence Diagram and Class Diagram which provides a visual representation of the requirements, logical flow and their class representations.

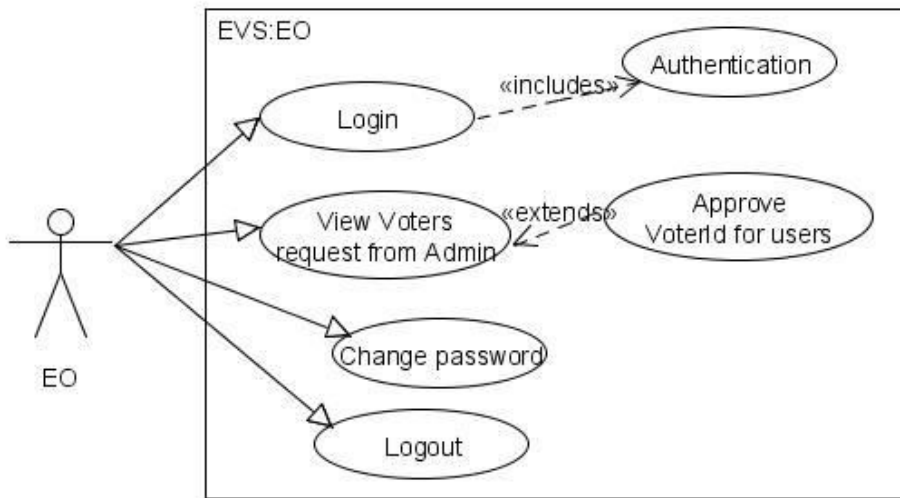
3.1 Use Case Diagrams

The requirements of a system can be represented using a use case model in the Use Case Diagram. The use case diagram for the actors of this case study is given as below.

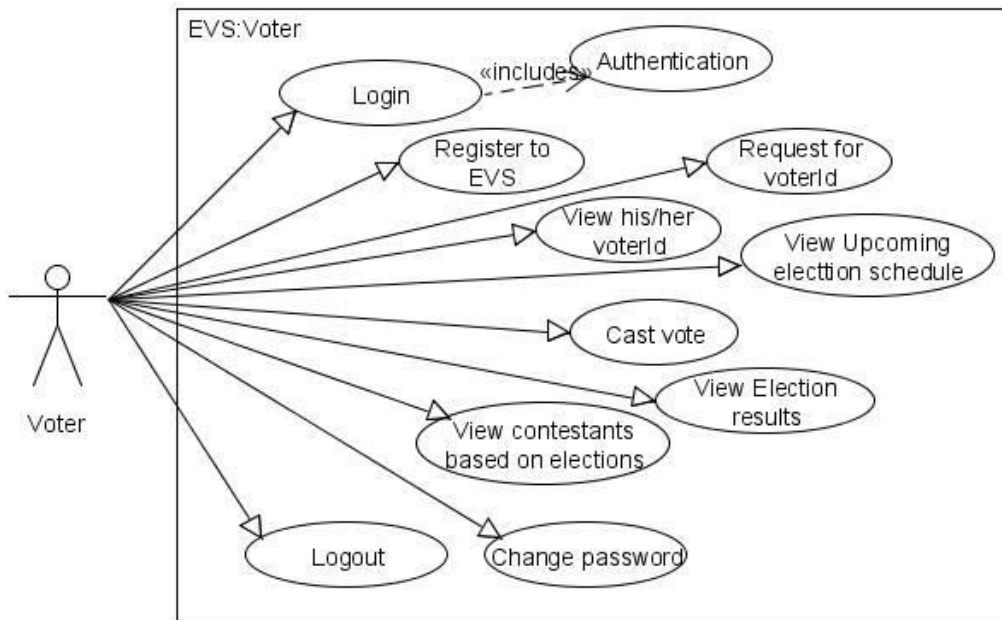
3.1.1 Use Case Diagram for Admin



3.1.2 Use Case Diagram for Electoral Officer



3.1.3 Use Case Diagram for Voter



3.2 Use case Definition

Generally, in a design document, Use case definitions should be written for all the *Requirements* of the system.

Note: Participants are expected to document use case definitions for all requirements. However, for few requirements documented below for reference.

Below table explains 'Use Case' definition for requirement "AA-001" - Login operation for all users.

3.2.1 Login

USE CASE #	AA-001 <i>Login</i>	
Goal	<i>All users logging into the system should be authenticated using a unique login-id and password (operations to be supported based on type of user)</i>	
Preconditions	If the user type is 'Admin' or 'Electoral Officer', credential details should exist. If the user type is 'Voter', he/she should be registered.	
Success End Condition	If the user type is 'Admin', then redirect to the Admin page. If the user type is 'Electoral Officer', then redirect to the Electoral Officer page. If the user type is 'Voter', then redirect to the Voter page.	
Failed End Condition	<i>The end user is redirected to an Error Page, and/or is asked to re-enter login credentials.</i>	
Primary, Secondary Actors	Admin, Electoral Officer, Voter.	
Trigger	<i>Login button</i>	
DESCRIPTION	Step	Action
	1	<i>Enter Login credentials (id & password)</i>
	2	<i>Click on Login button</i>
	3	<i>If id & password is Success, then identify user type Display appropriate(Admin/EO/voter) home page</i>
	Step	Branching Action
	1	<i>If 'id' is not existing then return with requesting for registration</i>
	2	<i>If password is not matching return with suitable error message say 'reenter id & password'</i>
Related Information/Use cases	<i>Not Applicable</i>	
Priority	<i>P1</i>	
Performance	<i>Approx. in seconds</i>	
Frequency	<i>Approx. few users per minute</i>	
Assumptions	Admin/EO login credentials are available in the database and others are already registered with their credentials	

Below table explains 'Use Case' definition for requirement "AD-001" – ADD Election Details operation for Admin user only.

3.2.2 ADD Election Details

USE CASE #	AD-001 <i>Add Election details</i>	
Goal	<i>To enable Administrator to create and add new Election</i>	
Preconditions	Administrator must be logged in to be able to create a new <i>Election</i> .	
Success End Condition	<i>"Redirect to Admin home page"</i>	
Failed End Condition	<i>"Redirect to Error Page"</i>	
Primary, Secondary Actors	<i>Administrator</i>	
Trigger	<i>Create button</i>	
DESCRIPTION	Step	Action
	1	<i>Provide appropriate election details</i>
	2	<i>Click on Add election button</i>
	Step	Branching Action
	1	<i>If failed to add election details</i>
	2	<i>Display appropriate message to the Admin</i>
Related Information/Use cases	<i>Not applicable</i>	
Priority	<i>P1</i>	
Performance	<i>Approx. 4 sec</i>	
Frequency	<i>40 / month</i>	
Assumptions	<i>Admin/EO login credentials are available in the database</i>	

3.3 Class Diagram

The class diagram is a very basic concept in object-oriented world. Class diagrams demonstrate a model, describing what attributes and behavior it has rather than describing the methods for accomplishing operations. Class diagrams are very useful in representing relationships between classes and interfaces.

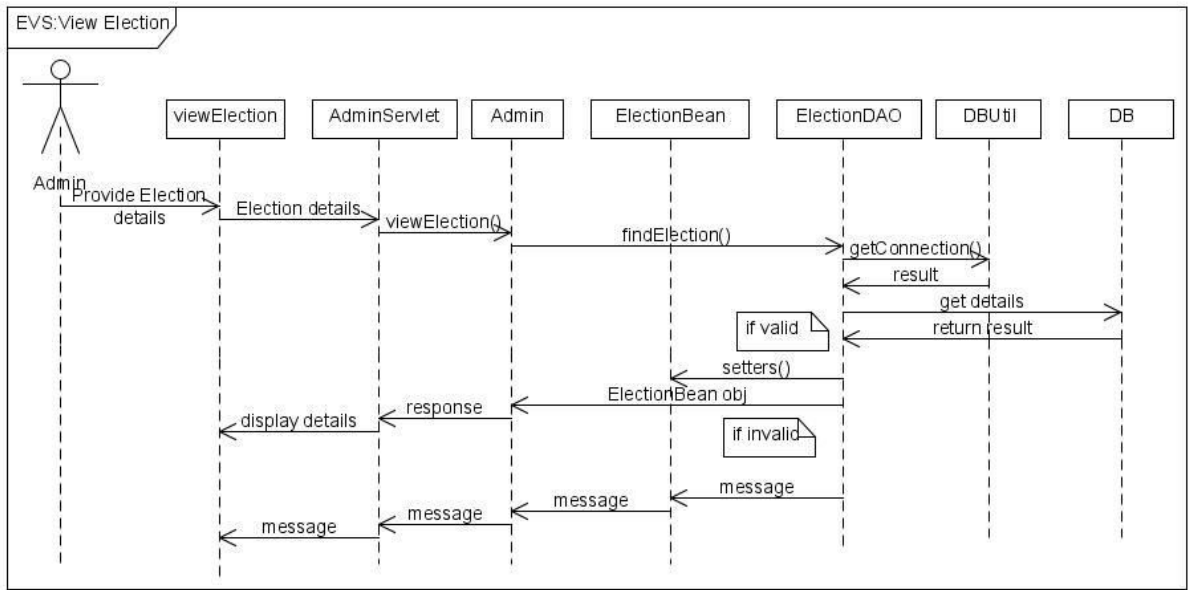
<class diagram to be drawn by RLL participants>

3.4 Sequence Diagram

A graphical representation of a module's function invoking functions of other modules in order to achieve a task (specific user requirement) is called a sequence diagram. A sequence diagram for the authentication process is given below for reference. The below example is for a Web Application using servlet/jsp.

3.4.1 Add Election details:

3.4.2 View Election details (AD-003)



3.5 Packages / Classes / Interface

This section provides a brief outlook on the packaging hierarchy along with the respective classes to be used for the implementation.

The 4 packages mentioned below are for both GUI and Web Application.

Packages	
Package	Description
com.Int.evs.service	This package contains all the Service classes
Com.Int.evs.bean	This package contains all the bean classes
com.Int.evs.dao	This package contains all the DAO functionality classes
com.Int.evs.util	This package contains all the generic functionality classes

This package is used only for a GUI application.

com.Int.evs.ui	This package contains all the UI related classes [For Core Java]
-----------------------	--

The package for the controller class should be used as below based on the type of application

com.Int.evs.listener or	This package contains all the controller classes	
com.Int.evs.servlet or	listener - core java	
com.wiro.evs.action or	servlet - Web Applications	
com.wiro.evs.controller	action - Struts	
	controller - Spring	

Package com.Int.evs.bean

Class Name	Attributes	Data Type
ProfileBean	userID	String
	firstName	String
	lastName	String
	dateOfBirth	Date
	gender	String
	street	String
	location	String
	city	String
	state	String
	pincode	String
	mobileNo	String
	emailID	String
	password	String

Detailed Design

CredentialsBean	userID	String
	password	String
	userType	String
	loginStatus	int
ElectionBean	electionID	String
	name	String
	electionDate	Date
	district	String
	constituency	String
	countingDate	Date
PartyBean	partyID	String
	name	String
	leader	String
	symbol	String
CandidateBean	candidateID	String
	name	String
	electionID	String
	partyID	String
	district	String
	constituency	String
	dateOfBirth	Date
	mobileNo	String
	address	String
	emailID	String
ApplicationBean	userID	String
	constituency	String
	passedStatus	int
	approvedStatus	int
	voterID	String

Detailed Design

ResultBean	serialNo	int
	electionID	String
	candidateID	String
	voteCount	int
EOBean	electoralOfficerID	String
	constituency	String

Package com.Int.evs.service

Interface Summary	
Interface	Description
Administrator	<p>Entity class for Administrator dealing with the admin process functionalities</p> <p>Method Summary</p> <p>String addElection(ElectionBean electionBean) Return value must be either: "SUCCESS", "FAIL", "ERROR"</p> <p>ArrayList<ElectionBean> viewAllUpcomingElections()</p> <p>ArrayList<ElectionBean> viewElections()</p> <p>String addParty(PartyBean partyBean) Return value must be either: "SUCCESS", "FAIL", "ERROR"</p> <p>ArrayList<PartyBean> viewAllParty()</p> <p>String addCandidate(CandidateBean candidateBean) Return value must be either: "SUCCESS", "FAIL", "ERROR"</p> <p>ArrayList<CandidateBean> viewCandidateDetailsByElectionName(String electionName)</p> <p>ArrayList<ApplicationBean> viewAllAdminPendingApplications()</p> <p>boolean forwardVoterIDRequest(String userId)</p> <p>ArrayList<CandidateBean> viewCandidateDetailsByParty(String partyId)</p> <p>Map approveElectionResults(String electionId) Note: Suitable data to be returned from Election and Result tables</p>

ElectoralOfficer	<p>Entity class for Electoral Officer for dealing with the EO process functionalities</p> <p>Method Summary</p> <p>String generateVoterId(String userId, String status) Return value must be either: "SUCCESS", "FAIL", "ERROR" Note: VoterId should be first 2 letters of user First Name with 2 letters constituency name followed by 4 digits auto generated number</p> <p>ArrayList<ApplicationBean> viewAllVoterIdApplications()</p>
Voter	<p>Entity class for Voter dealing with the voter process functionalities</p> <p>Method Summary</p> <p>String castVote(String userId, String electionId, String candiadId) Return value must be either: "SUCCESS", "FAIL", "ALREADY CASTED"</p> <p>ArrayList<CandidateBean> viewCandidatesByElectionName(String electionName, String constituency)</p> <p>ArrayList<ResultBean> viewListOfElectionsResults()</p> <p>String requestVoterId(String userId) Return value must be either: "SUCCESS", "FAIL", "APPLIED", "REJECTED" Note: Applied-> Application is pending with EO or Admin, Rejected-> Application rejected by EO</p> <p>String viewGeneratedVoterId(String userId, String constituency) Return value must be either: "<8 character voterId>", "FAIL", "PENDING", "REJECTED"</p> <p>ArrayList<ElectionBean> viewListOfElections()</p>

Package com.int.evs.dao

Find below the suggestive approach for CRUD operations [method naming & signature] for the DAO Interface/classes. Create the necessary DAO classes.

Interface Name	Description
xyzDAO	<p>DAO interface/class to deal with operations related to the specific table.</p> <p>Method Summary</p> <p>String createXYZ(BeanoObject)</p> <p>int deleteXYZ(ArrayList<String>)</p> <p>boolean updateXYZ(BeanoObject)</p> <p>BeanoObject findById(String)</p> <p>ArrayList<BeanoObject> findAll()</p>

- If required, additional find methods can be created.

Package com.Int.evs.util

Interface Summary	
Interface	Description
Authentication	This class is responsible for performing the Authentication and Authorization process.
	Methods boolean authenticate (CredentialsBean user) String authorize (String userId) boolean changeLoginStatus (CredentialsBean user, int loginStatus)
DBUtil	This class is responsible for the Database connection establishment.
	Methods static Connection getDBConnection (String driverType)
User	Entity class for handing different types of users Method Summary String login (CredentialsBean credentialsBean) Return value must be either: "A", "V", "E", "FAIL", "INVALID" A->Admin, V-> Voter, E->Electoral Officer Wrong username/password should return INVALID. boolean logout (String userId) String changePassword (CredentialsBean , String) Return value must be either: "SUCCESS", "FAIL", "INVALID" String register (ProfileBean profileBean) Return value must be either: <userId of lenght 6>, "FAIL" Note: userId-> first 2 letter of first name followed by 4 digit auto generated number

Note: Wherever empty or NULL is the response in all such cases suitable message has to be displayed for user.

3.6 UI Templates

3.6.1 UI Principle

The UI [Presentation Layer] should be designed with the below mentioned principles which helps easy interaction by the user to the application.

3.6.2 UI controls and Usage Principle

UI Type	Controls	Description
Direct Entry	Text Box, Text Area	Any input that cannot be predicted and needs the user to key in. e.g Name, Address, contact no etc.
Static Selection	Option Button, Check Box, Drop Down	Should be used where the input can be predefined. e.g gender, month [Jan – Dec] etc. If number of items is more, drop down is preferred.
Dynamic Selection	Drop Down	The items for the drop down should be retrieved from a stored data. e.g Displaying Districts in a drop down from places table.
Automation	Label Text Field [Read Only]	Data's that are calculative or an output of a function. e.g : Displaying system date, showing total amount etc.
Decision Control	Button	Operations like submit, save, clear should be executed only upon clicking respective buttons.

3.6.3 UI Template

This section contains the design template for the website home page [Fig. 1] that will be displayed at the time of opening this web application and Actor specific home page [Fig. 2].

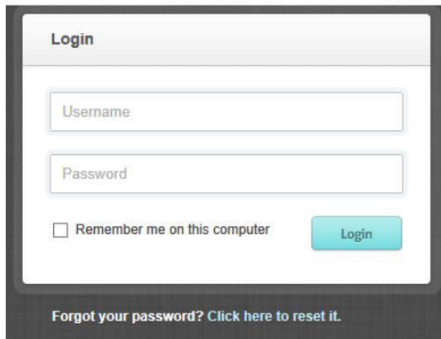
<logo>	< Project Title >
<div> About Us Contact Us </div>	
<div>< General Info ></div>	<div>  <p>New User? Register Here</p> </div>
Copyright @ 2013 Int Technologies. All rights reserved	

Fig. 1 - Main Page [First Page to open]

<logo>	< Project Title >		
< Logged in Name >		Home	Logout
<Navigation Links>	< Page based on the navigation link selected>		
<Navigation Links>			
<Navigation Links>			
<Navigation Links>			
<Navigation Links>			
<Navigation Links>			
Copyright @ 2013 Int Technologies. All rights reserved			

Fig. 2 - Home Page for Actor

<logo>	< Project Title >						
< Logged in Name >				Home Logout			
< Title for the View Screen >							
<Col Head>	<Col Head>	<Col Head>	<Col Head>	<Col Head>	<Col Head>		
						Edit	Delete
						Edit	Delete
						Edit	Delete
						Edit	Delete
						Edit	Delete
						Edit	Delete
						Edit	Delete
Copyright © 2013 Int Technologies. All rights reserved							

Fig. 3 – View Screen with Edit and Delete Functionality

4. Critical Functions and Focus for Testing

Authorization & Authentication are the critical functions need to be implemented before performing the tasks.

5. Limitations

- Each user will hold only one voter-ID and cast vote only once for a particular election
- The Electoral Officer and Administrator are also users who can cast vote
- The number of constituencies is limited to 5
- One electoral officer is assigned to one constituency
- Only one election can be scheduled on a date

6. APPENDIX

1. Table: EVS_TBL_User_Credentials

This table contains Authentication Information for Administrator, Electoral Officer & Voter

Field Name	Data Type	Description
Userid	VARCHAR2(6)	Primary Key*
Password	VARCHAR2(20)	Not Null
Usertype	VARCHAR2(1)	Either ['A','E','V']
Loginstatus	NUMBER(1)	Either [1 ,0]

* First 2 letters of First Name followed by 4 digits auto generated number

2. Table: EVS_TBL_User_Profile

This table contains User specific details entered during User Registration.

Field Name	Data Type	Description
Userid	VARCHAR2(6)	Foreign Key
Firstname	VARCHAR2(15)	Not Null
Lastname	VARCHAR2(15)	Not Null
Dateofbirth	DATE	Not Null
Gender	VARCHAR2(7)	Not Null
Street	VARCHAR2(30)	Not Null
Location	VARCHAR2(15)	Not Null
City	VARCHAR2(15)	Not Null
State	VARCHAR2(15)	Not Null
Pincode	VARCHAR2(10)	Not Null
MobileNo	VARCCHAR(10)	Exact 10 digit only
EmailId	VARCHAR2(30)	

* First 2 letters of First Name followed by 4 digits auto generated number

3. Table: EVS_TBL_Election

This table contains Election specific information, as necessary and applicable.

Field Name	Data Type	Description
ElectionId	VARCCHAR2 (6)	Primary Key
Name	VARCHAR2(15)	Not Null
Electiondate	DATE	Not Null
District	VARCHAR2(15)	Not Null
Constituency	VARCHAR2(15)	Not Null
Countingdate	DATE	Not Null

4. Table: EVS_TBL_Party

This table contains Party details like Party Name, Symbol & Leader Name.

Field Name	Data Type	Description
PartyId	VARCHAR2 (6)	Primary Key
Name	VARCHAR2(20)	Not Null
Leader	VARCHAR2(20)	Not Null
Symbol	VARCHAR2(40)	File reference

5. Table: EVS_TBL_Candidate

This table contains candidate details like Name of the Candidate, Election Name, Party Name and the contact details.

Field Name	Data Type	Description
CandidateId	VARCHAR2 (6)	Primary Key
Name	VARCHAR2(20)	Not Null
ElectionId	VARCHAR2 (6)	Foreign Key
PartyId	VARCHAR2 (6)	Foreign Key
District	VARCHAR2(20)	Not Null
Constituency	VARCHAR2(20)	Not Null
Dateofbirth	DATE	Not Null
MobileNo	VARCHAR(10)	Exact 10 digit only
Address	VARCHAR2(50)	Not Null
EmailId	VARCHAR2(20)	May or may not Null

6. Table: EVS_TBL_Application

This table contains details about request of voterid by user and the request status.

Field Name	Data Type	Description
Userid	VARCHAR2 (6)	FK
Constituency	VARCHAR2 (20)	Not Null
Passedstatus	NUMBER(2)	Either [1 2 3]*
Approvedstatus	NUMBER(2)	Either [0 1]**
VoterId	VARCHAR2(8)	***

* 1-> New request, 2->Admin forwards to EO, 3-> EO approves/rejects

** 0-> EO rejects, 1-> EO approve

*** 2 letters of user first name followed by 2 letters constituency name followed by 4 digit auto generated number

7. Table: EVS_TBL_Result

This table contains the details about the no of vote received by each candidate.

Field Name	Data Type	Description
Serialno	NUMBER(6)	PK, Autogenerated
Electionid	VARCHAR2 (6)	FK
Candidateid	VARCHAR2 (6)	FK
Votecount	NUMBER(5)	Not Null

8. Table: EVS_TBL_EO

This table contains the details about the electoral officer.

Field Name	Data Type	Description
Electoralofficerid	VARCHAR2 (6)	Primary Key
Constituency	VARCHAR2(25)	Not Null

9. Table: EVS_TBL_Voter_Details

This table contains the details about the voter id, election id & corresponding candidate id.

Field Name	Data Type	Description
Serialno	NUMBER(6)	PK, Autogenerated
Candidateid	VARCHAR2 (6)	FK
Electionid	VARCHAR2 (6)	FK
Voterid	VARCHAR2 (8)	FK

Database Sequences

Sequence Name	Purpose	Start With
evs_seq_userid	User ID	1000
evs_seq_requestId	Voter Request ID	1000
evs_seq_electionId	Election ID	1000
evs_seq_partyId	Party ID	1000
evs_seq_candidateId	Candidate ID	1000
evs_seq_voterid	Voter ID	1000
evs_seq_resultId	Result ID	1000