FUNCTIONAL SPECIFICATION

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
1.00	19-Sep-2011	Initial Draft			
2.10	Sept-2013	Revision	Mapping with CPC Tool		
2.20	Nov-2013	Revision	Aligning with UCF		

Affected Groups

Development Engineering
Quality Assurance
XYZ Automation Ltd.

List of Reference Documents

Name	Version No.
1. Request For Proposal	1.1
2.	
3.	
4.	

Prepared by/Date **Reviewed by/Date** **Approved by/Date**

Table of Contents

1. INTRODUCTION 3

2. SYSTEM OVERVIEW 3

3. SUB-SYSTEM DETAILS 4

4. DATA ORGANIZATION 5

5. ASSUMPTIONS 6

6. EXPECTATIONS 6

7. ACCEPTANCE CRITERIA 7

8. TRACEABILITY TO REQUIREMENTS 7

9. ACRONYMS AND GLOSSARY 7

1 Introduction

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

XYZ Automation Ltd recently planned to develop “Electronic Voting System” - a standalone/Web application [Core Java Batches - Swing Application; J2EE Batches - Web Application] to automate the process of planning and managing the voting system, as well as the actual voting activity.

Scope and Overview:

The scope of the Electronic Voting System (EVS) will be to provide the functionality as described below. The system will be developed on a Windows operating system using Java/J2EE.

2 System Overview

The Electronic Voting System should support basic functionalities (explained in section 2.1) for all below listed users.

- Administrator (A)
- Electoral Officer (E)
- Voter (V)

2.1 Authentication & Authorization

2.1.1 Authentication:

Any end-user should be authenticated using a unique login ID and password.

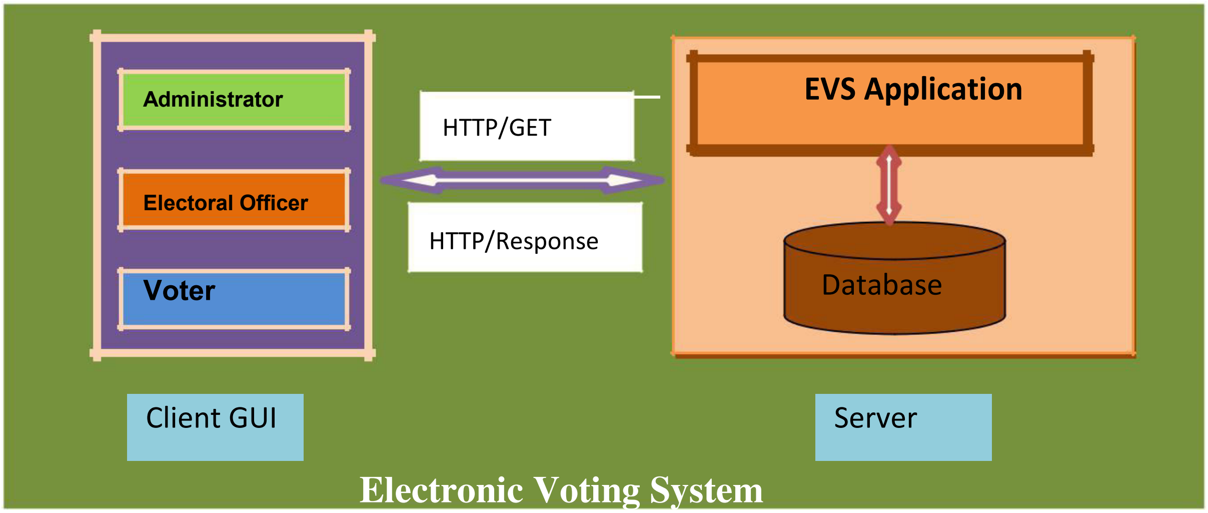
2.1.2 Authorization

The operations supported and allowed would be based on the user type. For example, Administrator has the rights to add/modify/delete election details and candidate details.

2.2 Functional Flow

The functional flow of the messages across different application components is shown below.

Ex. - Web Application.



2.3 Environment

The system will be developed on a Windows XP machine using J2EE, JSP/HTML, and JDBC.

- Intel hardware machine (PC P4-2.26 GHz, 512 MB RAM, 40 GB HDD)
- Server – Apache Tomcat 6 or higher
- Database – Oracle 9i or higher
- JRE

- Eclipse IDE

3 Sub-system Details

The Electronic Voting System (EVS) is defined with three types of users (Administrator, Electoral Officer & Voter), wherein all users need to login successfully before performing any of their respective operations.

Find below (section 3.1 & 3.2) tables that provides functionality descriptions for each type of user / sub-system. Against each requirement, indicative data is listed in column 'Data to include'. Further, suggested to add/modify more details wherever required with an approval from customer/faculty.

3.1 Administrator

The administrator as a user is defined to perform below listed operations after successful login.

ID	Objects	Operations	Data to include	Remarks
AD-001 to AD-003	Election	Add View	Election Name, State, Date, Constituency etc.	ElectionId should be auto generated on adding election details
AD-004 to AD-005	Party	Add View	Name, Leader, Symbol	Party symbol should be an image file PartyId should be auto generated
AD-006 to AD-007	Candidate	Add/Assign View	Name, Age, Address, Party, Constituency, Contact details etc.	CandidateId should be auto generated
AD-008	VoterRequest	View Approve	Name, Request Id, District, Constituency, voter-id, application status etc.	
AD-009	Candidate, Party	View	Candidate details and Party details	
AD-010	Election	Declare/Approve Results	Party details, Candidate details, No. of votes, etc	No. of votes to candidates and party details should be shown in ascending /descending order

3.2 Electoral Officer

The **Electoral Officer** as a user is defined to perform below listed operations after successful login.

ID	Objects	Operations	Data to include	Remarks
EO-001 to EO-002	VoterRequest	View Approve/Generate voter-id	Name, Request Id, District, Constituency, voter-id, application status etc.	1. Only requests approved by admin should be visible 2. EO has the option to either "Accept " or "Reject" for each user to be provided . 3. On **accept**, the voter-Id gets auto generated and stored in the respective voters

				table

3.3 Voter

The ***Voter*** as a user is defined to perform below listed operations after successful login.

ID	Objects	Operations	Data to include	Remarks
US-001	UserProfile	Register	Name, Date of Birth, Gender, Address, Mobile No District, etc.	UserId should be auto generated
US-002 to US-003	VoterRequest	Add View voter-id	UserId, Constituency etc.	User should be registered for a particular constituency only. VoterId should be auto generated
US-004	Schedule	View	Election, DateOfElection, etc.	Only upcoming elections to be visible
US-005	Candidate	View	Election details, Candidate Name, Party, etc	Election-wise candidates should be visible
US-006 to US-007	Election	Cast Vote View Result	ElectionId, Date, Candidate Name etc.	Results should be based on Election name, constituency and date

[Swing Application - Core Java]

- US-007 : Allow user to view results for multiple constituencies simultaneously.
  - Hint: Use multithreading.
- US-007 : Allow user to generate election results in HTML format.

[Web Application - J2EE]

- US-006 : Use | Create Web services for casting vote.
- US-007 : Allow user to generate election results in PDF format.

NOTE:

- The users (voter) should have the ability to cast their vote for an election declared on a date, to one of the contesting parties in their own constituencies.
- The voter should be blocked from casting his/her vote twice on the same election.
- The Election list should have max 10 (recent) election dates for which, counting has already being done.
- The election results should be displayed in a sorted order (descending) based on No of Votes.

1.1 Login | Logout

[Swing Application - Core Java]

- Use System properties to enable the application to Startup with default/last user details for login.
- Enable the application to run from command prompt with user credentials.

[Web Application - J2EE]

- Implement Session tracking for all logged in users before allowing access to application features. Anonymous users should be checked, unless explicitly mentioned.

4 Data Organization

This section explains the data storage requirements of the Electronic Voting System and **indicative data description** along with suggested table (database) structure. The following section explains few of the tables (fields) with description. However in similar approach need to be considered for all other tables.

1.2 Table: UserProfile

The user specific details such as name, address, authentication and authorization / privileges should be kept in one or more tables, as necessary and applicable.

Field Name	**Description**
UserID	Customer ID is auto generated after registration and it is used as LoginID.
Name	Customer Name [first name & last name]
DOB	DOB of Customer
Gender	Gender of user [Male / Female]
PresentAddress	Present Address of Customer
PermanentAddress	Permanent Address of Customer
PhoneNumber	10 digit contact Number
Emailid	Email ID of the traveler

1.3 Table: UserCredentials

The table contains Authentication Information for Administrator, Electoral Officer and Voter

Field Name	**Description**
UserType	Administrator and Customer
UserID	User Identification, corresponding to UserProfile table
Password	Password
LoginStatus	Login status of the user

1.4 Table: ElectionDetails

This table contains information related to the credit cards of the users used for making payments.

Field Name	**Description**
Id	Unique Id for the election
Name	Name for the election viz. Assembly Election - 2013
Date	Date of election
District	District of election
Constituency	Constituency of election
CountingDate	Date of counting votes and/or declaring results for the election

5 Assumptions

- User Interface: The type of client interface (front-end) to be supported– GUI based/Web based
- Each user will hold only one voter-ID and cast vote only once for a particular election
- The scope of the application is limited to only one country.
- Information about state, district and constituency may be already stored in database.

6 General Expectations

- The server should be a concurrent server servicing multiple clients
- Database can be implemented using Oracle 9i or above
- To begin with, the application should support at least 1 admin and 2 customers.
- Compilation and Build should be done using Eclipse IDE
- Source-code and all documents must be maintained (checked-in) in configuration management system (subversion)
- Int’s coding standards (for Java) should be followed,
- Deliverables should include compiled and tested source code, Unit Test Code (Using IntUT), IntStyle report and System test-plan / report documents.

NOTE:

1. **Validation of user Data**[1]
   - ✓ Struts 2 validation via XML or annotations or Spring MVC using JSR-303 annotations
   - ✓ AJAX validation without forcing the page to reload (Wherever applicable)
   - ✓ JavaScript validation (if necessary)
   - ✓ In case of Swing applications, use 'ClassInputVerifier' for validation
2. **UI Design –(for Web Application) Use DIV/CSS to control the style and layout**
3. **Create at least one SQL DML-statement inside PL/SQL blocks**

7 Acceptance Criteria

All P1 requirements have to be mandatorily implemented

8 Traceability to Requirements

Appropriate requirements from RS and FS are mapped here.

Document Reference ID & Description: (Doc ID from which this document is derived)

Sl. No.	Reference document: RS Requirement/Feature (Section ID/Name)	Current document: FS Location (Section ID/Name)
1.		
2.		

9 Acronyms and Glossary

Acronym and glossary for this document mentioned in the below table.

Abbreviation	Remark
EVS	Electronic Voting System
RS	Requirement Specification
FS	Functional Specification

[1] Validations should be performed at all levels of application appropriately.