1. **INTRODUCTION**

**1.1 Rationale**

“Online Voting System” is a web based application. This software is used to manage the election procedure through an online platform which is much more efficient than the existing system. This application uses an online database, which is accessible anywhere with the help of the and can produce results in real time. Votes are casted through an online website. Security of the application depends on voter-id.

**1.2 Problem Definition and Proposed solution**

**Problem Definition:**

The traditional ways of voting during elections includes ballot paper voting system and Electronic voting machines (EVM’s). In the traditional Ballot Paper voting system, the votes were cast by the means of the papers in which the voter used to vote by marking the ballot paper with a rubber stamp on or nearby the symbol of the candidate as per personal choice, the voter then folds the ballot paper and put it in the ballot box that used to be kept safe in the watch of allotted officers and polling agents of the candidates.

Examples of unusual functioning of Electronic Voting Machines EVM’s:

1. Malfunctioning of EVM seen during several elections
2. EVM could not start
3. EVM got jammed and did not open on counting day
4. EVM stopped functioning during the poll/in between the polling process
5. When button of one candidate pressed, light on another candidate got flashed
6. Polling staff was not at all updated or aware about the security measures.

Both systems have its disadvantages that lead to necessity of new simple and more efficient system for voting during elections.

**Proposed Solution:**

To overcome such deficiencies a new system for voting is proposed: **ONLINE VOTING SYSTEM.** This system shall reduce the time spend making long queues at polling stations during voting. It shall also enable voters to vote from any part of globe since this is an online application available on internet. Cases of vote miscounts shall also be solved since at the backend of this system resides a well developed database using MYSQL that can provide the correct data once it’s queried. Since the voting process shall be open as early as possible, the voters shall have ample time to decide when and whom to vote for.

Features of “Online Voting System”:

1. Voters can vote from anywhere.
2. Votes can be viewed by the Admin at real-time.
3. The data gathered can be further analysed.
4. The feedback system is available for any trouble.
   1. **Objective and Scope**

The Objective of working on this project is to bring about a new voting system that is more efficient and hassle free. The key points include:

* Provision of improved voting services to the voters through fast, timely and convenient voting.
* Check to ensure that the members who are registered are the only ones to vote.
* Online voting system will require being very precise or cost cutting to produce an effective election management system

**Project Scope:**

It is focussed on studying the existing system of voting in INDIA and to make sure that the people vote is count, for fairness in the elective positions. This will also produce:

* Less effort and less Labour intensive.
* Increasing number of voters as individuals will find it easier and more convenient to vote.

**1.4 PROJECT ORGANIZATION**

**1.4.1 Software Process Model**

The software model used in our project is Incremental model.

The incremental model applies the waterfall model incrementally. The series of releases is referred to as “increments”, with each increment providing more functionality to the customers. After the first increment, a core product is delivered, which can already be used by the customer. Based on customer feedback, a plan is developed for the next increments, and modifications are made accordingly. This process continues, with increments being delivered until the complete product is delivered. The incremental philosophy is also used in the agile process model.

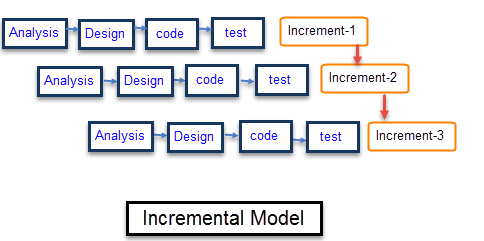


Figure 1.4 (a): - Incremental Model

**1.4.2 ROLES AND RESPONSIBILITIES**

* **ADMIN ROLES**
* Login using register Id.
* Check the results.
* Apply any data analysis needed.
* Maintain the database and system.
* **USER ROLES**
* Registration with valid e-mail address and voter-id.
* Login with the registered voter-id.
* Vote to the particular candidate.

**1.4.3 TOOLS AND TECHNIQUES**

The software tools used in our project are as follows:

* Netbeans
* MySQL server.
* Glassfish server.
* Mysql database.

**2. Existing System**

There is no such system exist in India. In India the only way to give our opinion is to visit our nearest pooling booth and stand in the long queues while waiting for your turn. There are also lots of security concerns which will also was definitely comes in voters mind such as booth capturing by anti social elements or vote selling under pressure and many more. Also the process which was in used right now was very slow as it takes almost ten minutes once the voter enters the voting room. There are several problems that our existing system put forwards to the voters And hence lots of people are unable to put their opinions on the board because of the system.

**3.** **System Analysis and Requirement Specification**

The purpose of SRS (Software Analysis Requirement Specification) document is to describe the external behaviour of the website. It defines the operations, performance and interfaces and quality assurance requirement of the website. The complete software requirements for the system are captured by the SRS (Software Analysis Requirement Specification).

**3.1 Functional and Non-Functional Requirements**

**3.1.1 Functional**

A functional requirement defines a function of a system or its components. A function is described as a set of inputs, the behaviour and outputs.

**Inputs:**

* User would provide his voter-id details.
* User would enter the OTP sent to the registered number.

**Processing:**

* Maintain Tables:

A function will run on server which will store the data into the database. The function will update the whole database and ensure a single vote on single voter-id.

Return the number of votes on the every candidate when requested by the admin.

**Output:**

Display the result obtained from server on result page and notify the user that the vote has been casted.

**3.1.2 Non-Functional**

**I. Performance Requirement**

Performance requirement is concerned with the speed of operations of functions and their accuracy. For optimum performance of application some non-functional requirements are recommended:

1. Good Internet Connectivity
2. Google Chrome Web Browser
3. A valid voter-id.

**II. Reliability**

High Reliability is the measure of how a product behaves in varying circumstances and our project is reliable because there are less chance of errors and exceptions and works well in varying circumstances. The probability that application will perform required function without failure would depend on traffic on server as well as the good internet speed.

**III. Scalability**

Scalability is the capability of a system, network, or process to handle a growing amount of work, or its potential to be enlarged to accommodate that growth. We have tried our best to make this website highly scalable. Fast access to users in reading and writing, to handle a good amount of traffic in future. Scalability is measured in functional, load, generation scalability

**IV. Security**

Security is the ability of the software to remain protected from unauthorized access. This includes both change access and view access. This application is well secured using multiple levels of security constraints. It would be taken care that only an authenticated person can access this location, by checking validated fields in the database.

**V. Performance**

Performance requirement is concerned with the speed of operations of functions and their accuracy. For optimum performance of application some non-functional requirements are recommended:

* Good Internet Connectivity
* Google Chrome Web Browser
* A valid voter-id.

**3.2. Software and Hardware Requirements**

**HARDWARE**

* Operating System: Android API 17 or higher
* Hard disk: 200 MB or higher
* RAM: 512 MB or higher
* Processor: 1GHz or higher

**SOFTWARE**

* Netbeans
* MySQL Database
* MySQL server
* Glassfish server

**4. System Design**

**4.1 Use Case Diagram**

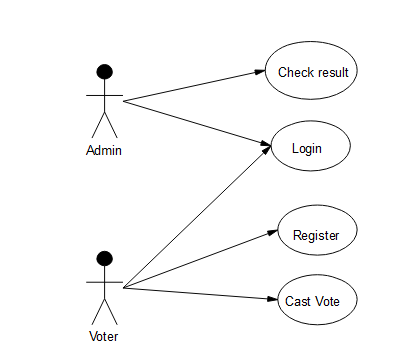


Figure 4.1: Use Case Diagram

**4.3 Activity Diagram**

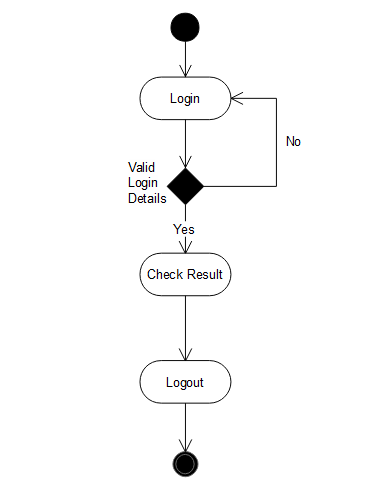
****

Figure 4.2 (a):- Activity Diagram Admin

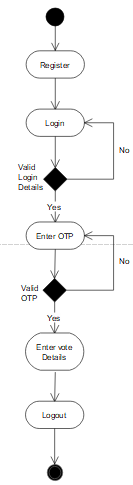


Figure 4.2 (b):- Activity Diagram User

**4.3 Sequence Diagram**

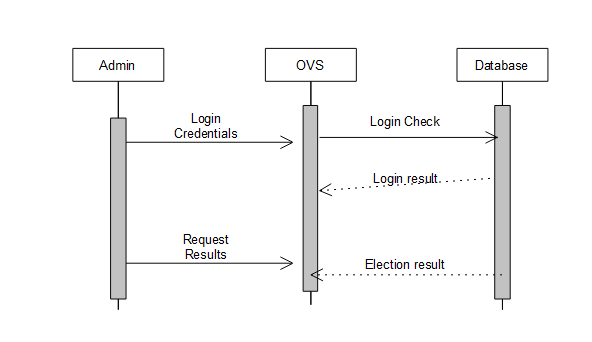
****

Figure 4.3(a): - Sequence Diagram Admin

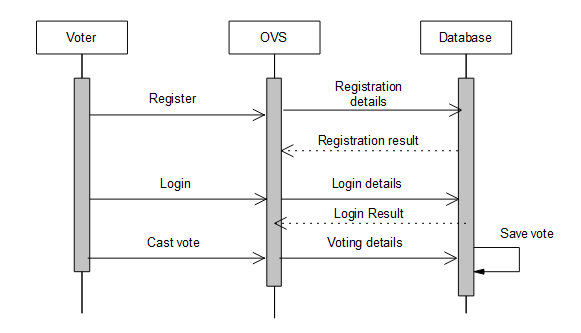


Figure 4.3(b): - Sequence Diagram voter

**4.4 Class Diagram**

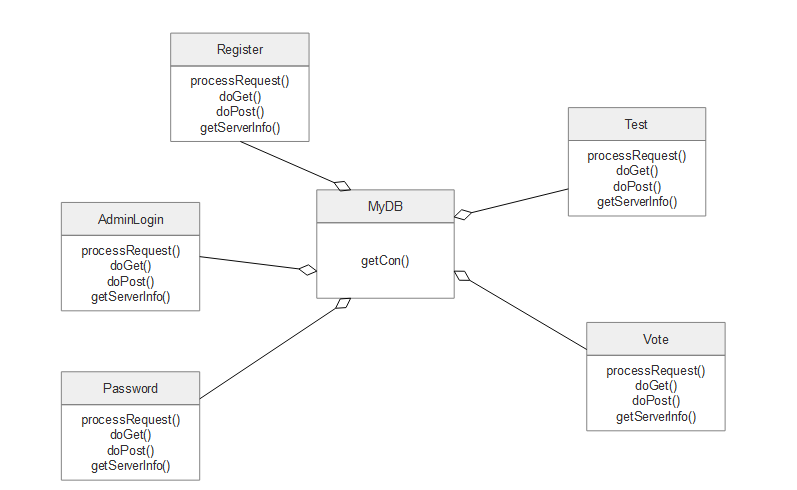
****

Figure 4.4: - Class Diagram

**4.5 Database Design**

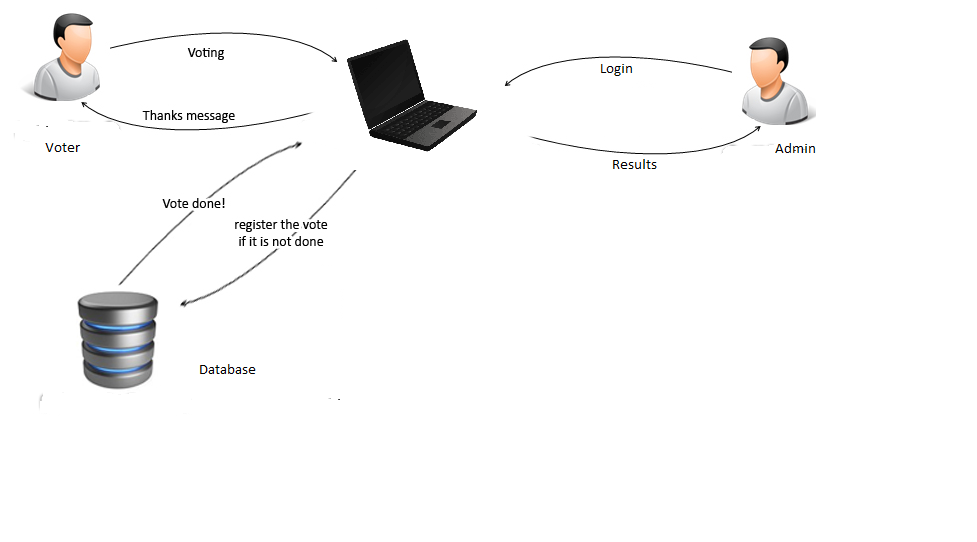


Figure 4.5: - Database Design

**4.6 Data Flow Diagram**

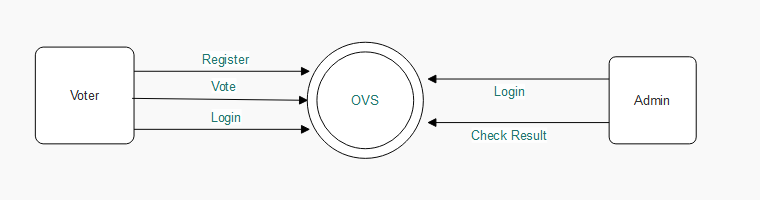
****

Figure 4.6: - Data Flow Diagram

1. **Project Implementation and Output Screens**

**5.1 Screen Images**

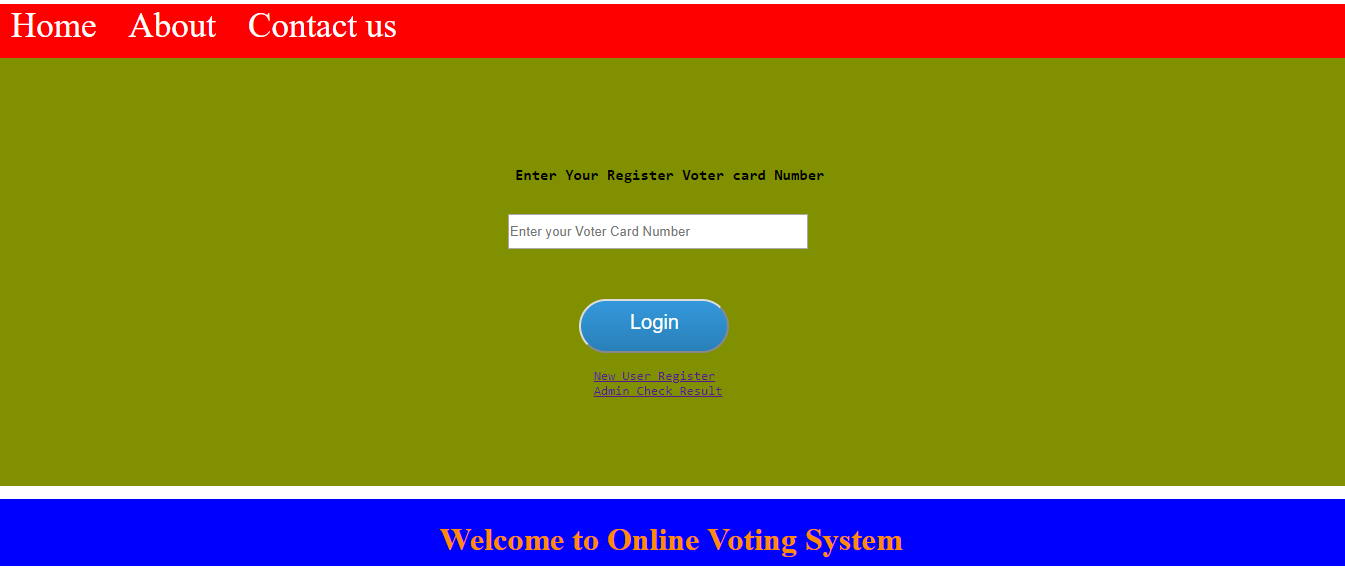
****

Figure 5.1 Home Screen

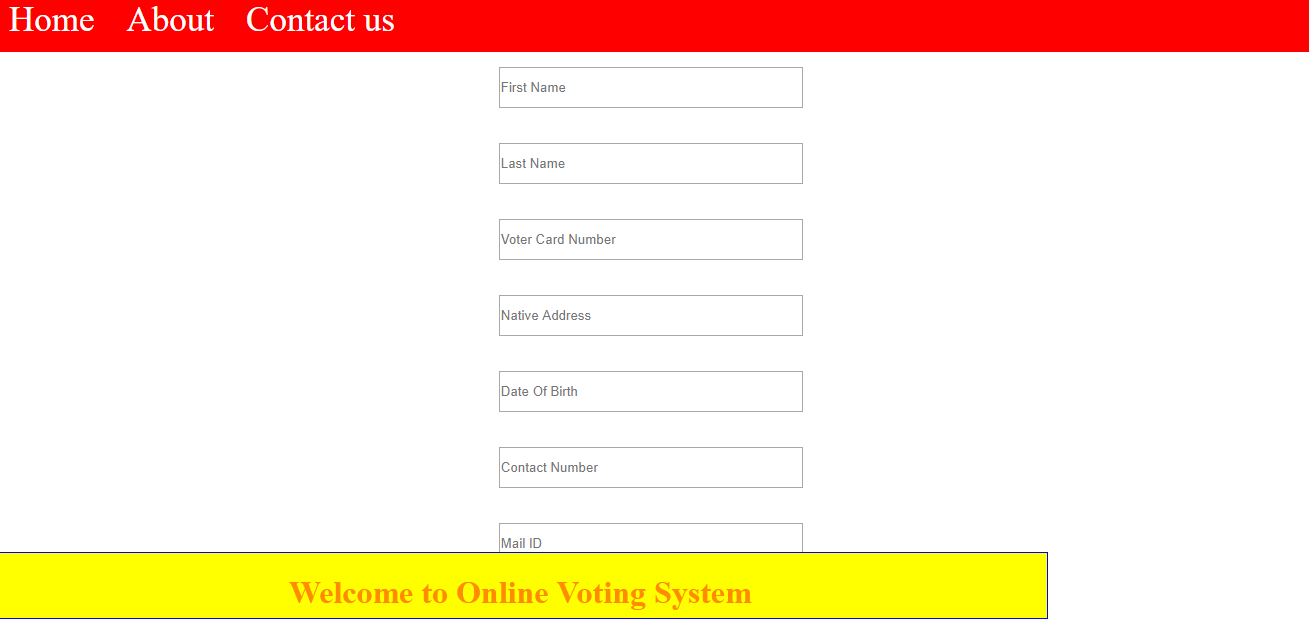


Figure 5.2 Registration Page



Figure 5.3 One Time Password Verification

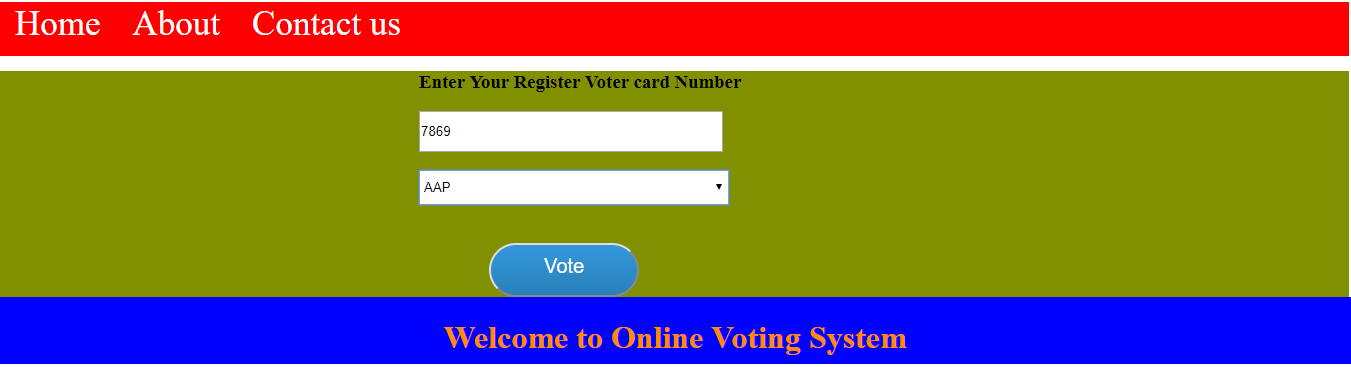


Figure 5.4 Voting Interface

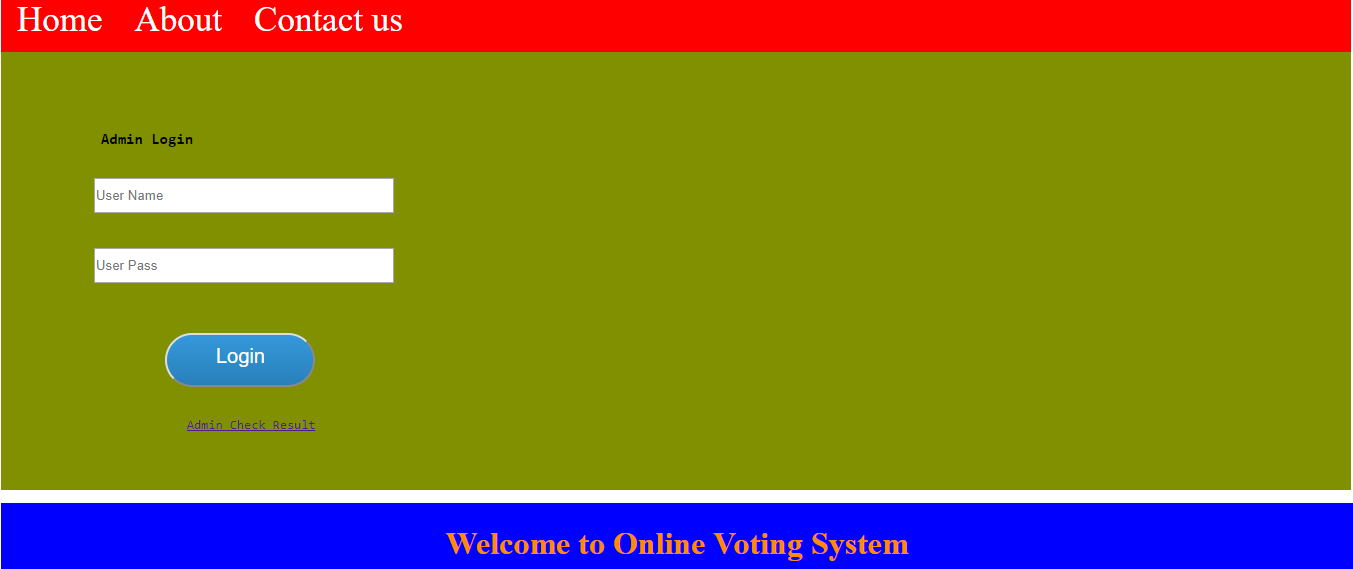


Figure 5.5 Administrator Login Page



Figure 5.6 Results of voting

**5.2 Important coding**

**Home Screen**

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html>

<head>

<title>Online Voting System</title>

<%@ include file="header.jsp"%>

<link href="css/home.css" rel="stylesheet" type="text/css">

<div class="back\_image">

<div class="reg" >

<form action="Vote" method="post">

<div class="check">

<h3 style="margin-left: 0px; margin-bottom: -20px;margin-top: 15px;">Enter Your Register Voter card Number</h3>

<br>

<input name="voter\_card\_number" placeholder="Enter your Voter Card Number" type="text">

</br>

<br>

<select style="height: 35px; width: 310px;" name="parties">

<option value="#">Select Your Party</option>

<option value="bjp">Bharatiya Janata Party</option>

<option value="app">AAP</option>

<option value="congrace">Congress</option>

</select>

</br>

<br>

<input value="Vote" type="submit" class="btn" style="padding-bottom: 40px; width: 150px; margin-left: 70px;">

</br>

</div>

</form>

</div>

</div>

</body>

<%@ include file="footer.jsp"%>

</html>

**User Registration**

import java.io.IOException;

import java.io.PrintWriter;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

import java.sql.Statement;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

public class Register extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

response.setContentType("text/html;charset=UTF-8");

try (PrintWriter out = response.getWriter()) {

String name = request.getParameter("name");

String surname = request.getParameter("surname");

String voter\_card\_number = request.getParameter("voter\_card\_number");

String address = request.getParameter("address");

String dob = request.getParameter("dob");

String contact = request.getParameter("contact");

String email = request.getParameter("email");

System.out.println("");

String s= "s";

MyDb db = new MyDb();

Connection con = db.getCon();

Statement stmt = con.createStatement();

stmt.execute("insert into voter\_register(name,surname,voter\_card\_number,contact,address,dob,email)values('"+name+"','"+surname+"','"+voter\_card\_number+"','"+contact+"','"+address+"','"+dob+"','"+email+"')");

out.println("registration success");

System.out.println("<script>alert('Register Success.')</script>");

response.sendRedirect("index.jsp?m="+s+"");

} catch (SQLException ex) {

Logger.getLogger(Register.class.getName()).log(Level.SEVERE, null, ex);

}

}

@Override

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

processRequest(request, response);

}

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

processRequest(request, response);

}

@Override

public String getServletInfo() {

return "Short description";

}

}

**Connection To Database**

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

import java.util.logging.Level;

import java.util.logging.Logger;

public class MyDb {

Connection con;

public Connection getCon()

{

try {

Class.forName("com.mysql.jdbc.Driver");

con = DriverManager.getConnection("jdbc:mysql://localhost:3306/voating", "root", "root");

} catch (ClassNotFoundException ex) {

Logger.getLogger(MyDb.class.getName()).log(Level.SEVERE, null, ex);

} catch (SQLException ex) {

Logger.getLogger(MyDb.class.getName()).log(Level.SEVERE, null, ex);

}

return con;

}

}

**6. Testing**

**6.1 Testing Strategy Adopted**

Various software-testing strategies have been proposed so far. All provide a template for testing. Things that are common and important in these strategies are: Testing begins at the module level and works “outward”: tests which are carried out are done at the module level where major functionality is tested and then it works towards the integration of entire system.

Different testing techniques are appropriate at different point of time: Under different circumstances, different testing methodologies are to be used which will be the decisive factor for software robustness and scalability. The developer of the software conducts testing and if the project is big then there is a testing team: All programmers should test and verify that their results are according to the specification given to them while coding. In cases where programs are big enough or collective effort is involved for coding, responsibilities for testing lies with the team as a whole.

A test approach is the test strategy implementation of a project, defines how testing would be carried out. Test approach has two techniques:

**Proactive -**An approach in which the test design process is initiated as early as possible in order to find and fix the defects before the build is created.

**Reactive -**An approach in which the testing is not started until after design and coding are completed.

There are many strategies that a project can adopt depending on the context and some of them are:

* Dynamic and heuristic approaches
* Consultative approaches
* Model-based approach that uses statistical information about failure rates.
* Approaches based on risk-based testing where the entire development takes place based on the risk
* Methodical approach, which is based on failures.
* Standard-compliant approach specified by industry-specific standards.

**6.2 System Testing**

System testing is the testing to ensure that by putting the software in different environments (e.g., Operating Systems) it still works. System testing is done with full system implementation and environment. It falls under the class of black box testing.

System Testing (ST) is a black box testing technique performed to evaluate the complete system the system's compliance against specified requirements. In System testing, the functionalities of the system are tested from an end-to-end perspective.

System Testing is usually carried out by a team that is independent of the development team in order to measure the quality of the system unbiased. It includes both functional and Non-Functional testing.

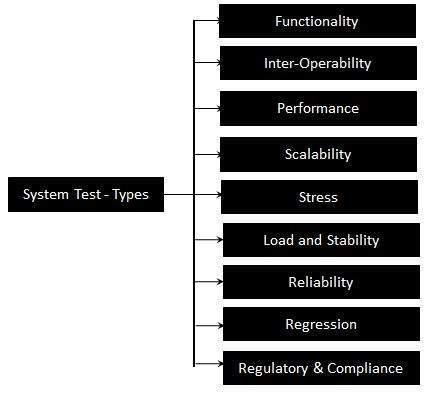


Figure 6.2 (a):- System Testing Types

**6.3 Unit Testing**

Unit testing is the testing of an individual unit or group of related units. It falls under the class of white box testing. It is often done by the programmer to test that the unit he/she has implemented is producing expected output against given input.

The following are the tests that are performed during the unit testing:

**Data Fetch from server using Async Methods: -** To check bus session and coordinates information fetched through server.

**Network Failures: -** To check network failures if any while data transmission between client and server.

**6.4 Test Plan & Test Cases**

Testing documentation involves the documentation of artifacts that should be developed before or during the testing of Software. Documentation for software testing helps in estimating the testing effort required, test coverage, requirement tracking/tracing, etc. This section describes some of the commonly used documented artifacts related to software testing.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case Number** | **Test Case** | **Description** | **Test Result** |
| T-001 | Login Module | That always a valid and registered user is able to login into the system.  Students can also login via Google Account. | Successful |
| T-002 | Voter Registration Module | Student registers through a valid email address and voter-id | Successful |
| T-003 | Voting system | Votes are casted. | Successful |
| T-004 | One time Vote | Votes are casted only once by a single voter-id | Successful |
| T-005 | Result check | The results are updated in real-time | Successful |

Table 6.4: Test cases

**7. Conclusion**

**7.1 Conclusion**

The conclusion of this study suggests that knowledge of specific domain improves the results. This Project has been implemented on web platform. Also, different attributes are added to project which are advantageous to the system. The requirements and specifications have been listed above. This project uses Mysql as backend infrastructure. Using different security system, the application will be unbiased and not manipulatable.

Due to simple MySQL structure, data can be easily updated in future whenever required.

**7.2 Future Enhancement**

* Encryption shall be done on both ends for security reasons.
* Aadhar API may be included for more security.
* Analysis of more captured data.
* More capturing of diverse date.
* More effective GUI.

**8. Appendix**

* Authentication: Process of establishing who you are.
* Authorization: Permission to access non-public information
* Client/server: A relationship in which client software obtains services from a server on behalf of a person.
* Crash: A computer system is said to crash when it stops working for some reason and must be restarted.
* Database: A collection of interrelated data values that may be integrated permanently into a single connected structure or integrated temporarily for each interrogation, known as a query.
* Database management system: A systematic approach to storing, updating, securing and retrieving information stored as data items, usually in the form of records in one or more files.
* Error message: A message that reports the detection of an error.
* Execute: To interpret a computer instruction and carry out the operations specified in the instruction.
* GUI: Graphical User Interface. Defines a format for scroll bars, buttons, menus, etc., and how they respond to the user.
* Module: A logically self-contained and discrete part of a larger computer program.
* SQL: Structured Query Language. ANSI standard data manipulation language used in most relational database systems. A language for requesting data from a relational database.
* User Id: A code that uniquely identifies a user and then provides access privileges to a computer system.

**8.1 Definitions, acronyms & abbreviations**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **ABBREVIATIONS** | **FULL FORM** |
| **1** | GUI | Graphical User Interface |
| **2** | E-MAIL | Electronic Mail |
| **3** | API | Application Programming Interface |
| **4** | XML | Extensible Markup Language |
| **5** | JDBC | Java Database Connectivity |
| **6** | SQL | Structured Query Language |
| **7** | HTML | Hyper Text Markup Language |

Table 8.1: Acronym and abbreviations

**8.2 References**

* [www.google.com](http://www.google.com)
  + Google search engine official website.
* http://sourceforge.net/
  + Largest repository of open source code and applications available on the internet.
* http://www.gnu.org
  + Website of the [Free Software Foundation](http://www.gnu.org/fsf/fsf.html) (FSF) which supports the open source community.
* http://www.thefreecountry.com
  + Large collection of good programming resources.
* http://www.1001tutorials.com/
  + Large collection of good tutorials for reference.
* http://www.codeguru.com/
  + Good articles on programming.