Software Requirements Specification (SRS)

Project: Citizen Management System

Technology: JSP, Servlets, JDBC

IDE: Eclipse

Java Version: JDK 8

Date: 11-08-2025

# 1. Introduction

## 1.1 Purpose

The purpose of the Citizen Management System is to provide an efficient and secure way to manage citizen records, including registration, updating, retrieval, and deletion of data. This system is intended for use by government authorities, municipal offices, and administrative staff.

## 1.2 Scope

The system will allow administrators to register new citizens, search citizens by ID, name, or location, update or delete citizen records, and ensure security through authentication. The system will use JSP for the front-end, Servlets for business logic, and JDBC for database interaction.

## 1.3 Definitions, Acronyms, and Abbreviations

JSP – Java Server Pages  
JDBC – Java Database Connectivity  
IDE – Integrated Development Environment  
ER – Entity Relationship  
CRUD – Create, Read, Update, Delete

# 2. Overall Description

## 2.1 Product Perspective

The Citizen Management System uses a three-tier architecture:  
- Presentation Layer: JSP  
- Business Logic Layer: Servlets  
- Data Layer: MySQL database via JDBC

## 2.2 Product Functions

• Citizen registration  
• Update citizen records  
• Delete citizen records  
• Search citizen records  
• Login authentication for admin

## 2.3 User Characteristics

Administrator: Knowledge of using a web application; responsible for managing records.  
Data Entry Operator: Limited to adding and viewing records.

## 2.4 Constraints

• Developed using JDK 8  
• Must run on Tomcat server  
• Uses MySQL database

# 3. Specific Requirements

## 3.1 Functional Requirements

FR1: The system shall allow admin login  
FR2: The system shall allow adding new citizens  
FR3: The system shall allow updating citizen details  
FR4: The system shall allow deleting citizen records  
FR5: The system shall allow searching citizens by name, ID, or location

## 3.2 Non-Functional Requirements

Performance: System should respond within 2 seconds for queries  
Security: User authentication required  
Scalability: Should support up to 10,000 records

# 4. Use Case Diagram

Actors:  
- Administrator  
- Data Entry Operator  
  
Use Cases:  
1. Login  
2. Add Citizen  
3. Update Citizen  
4. Delete Citizen  
5. Search Citizen  
(Use diagramming tools like draw.io or Lucidchart to create the visual.)

# 5. ER Diagram

Entities:  
- Citizen (CitizenID, Name, DOB, Gender, Address, Phone, Email, NationalID)  
- Admin (AdminID, Username, Password)  
  
Relationships:  
- Admin manages Citizen (1:N relationship)  
(Use diagramming tools to create the ER diagram.)

# 6. Testing Plan

## 6.1 Testing Types

• Unit Testing: Test individual JSP pages and Servlets  
• Integration Testing: Test JDBC connectivity with MySQL  
• System Testing: Verify the entire system works as per requirements  
• User Acceptance Testing (UAT): Confirm usability for end-users

## 6.2 Sample Test Cases

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case ID | Test Case Description | Input | Expected Output | Status |
| TC01 | Admin Login | Correct credentials | Dashboard loads | Pass |
| TC02 | Search Citizen | Name = 'John' | Records displayed | Pass |
| TC03 | Delete Citizen | CitizenID = 101 | Record removed | Pass |

# 7. Project Timeline

Phase | Duration | Tasks  
Requirement Gathering | Week 1 | Understand system needs  
Design | Week 2 | ER diagram, Use case diagrams  
Development | Week 3-5 | Coding JSP, Servlets, JDBC  
Testing | Week 6 | Unit, Integration, System testing  
Deployment | Week 7 | Final hosting

Gantt Chart Representation:  
Week: 1 2 3 4 5 6 7  
Req ███  
Design ███  
Dev █████████  
Test ███  
Deploy ███