GenAI Level 2 –Test Case Documentation

(AI-Powered Retrieval with Secure Role-Based Access)

Document Search Bot

Developed by

Ramaksha Kulkarni(27587)

1. Introduction

This document outlines the test cases created for validating the core functionality, user roles, API endpoints, and behavior of the Proof of Concept (POC) system. The application allows users to upload documents, interact with an AI model by asking questions based on the content of those documents, and enables admin-level operations such as managing documents and user roles. Testing is based on endpoints and flows implemented in the design.

2. Objective

To ensure that all implemented features and REST APIs behave as expected and produce the correct results across different user roles (Standard User and Admin). This includes verifying frontend and backend interaction, role restrictions, data extraction, AI response quality, and MySQL integration.

3. Test Scenarios and Cases

3.1 User Authentication & Role Verification

Test ID	**Scenario**	**API**	**Method**	**Expected Result**	**Role**
TC01	Signup with new user	/public/signup	POST	User is created in MySQL with role USER	Anonymous
TC02	Login with valid credentials	/public/login	POST	JWT token issued	User/Admin
TC03	Login with invalid credentials	/public/login	POST	Error response with 401 Unauthorized	-
TC04	JWT token required for access	/api/document	GET	403 Forbidden without token	-

3.2 Admin Document Management

Test ID	Scenario	API	Method	Expected Result	Role
TC05	View all documents	/api/document	GET	List of all uploaded documents returned	Admin
TC06	Upload a valid document	/api/document	POST	Document saved and processed successfully	Admin
TC07	Upload unsupported format	/api/document	POST	415 Unsupported Media Type	Admin
TC08	Upload oversized file	/api/document	POST	413 Payload Too Large	Admin
TC09	Delete document	/api/document/{filename}	DELETE	Document removed from storage	Admin

3.3 Admin User Management

Test ID	Scenario	API	Method	Expected Result	Role
TC10	View all registered users	/api/user	GET	List of users returned	Admin
TC11	Toggle user role	/api/user/{username}	GET	Role toggled between USER and ADMIN	Admin

3.4 Standard User AI Chat

Test ID	Scenario	API	Method	Expected Result	Role
TC12	Submit a valid question	/api/ask/{question}	GET	AI-generated answer based on document content	User/Admin
TC13	Submit question without upload	/api/ask/{question}	GET	Message: No file context available	User/Admin
TC14	Ask unrelated or vague question	/api/ask/{question}	GET	LLM returns informative fallback message	User/Admin

3.5 File Processing & Text Extraction

Test ID	Scenario	Action Description	Expected Result	Role
TC15	Upload DOCX with text	Upload via POST /api/document	Text extracted and stored temporarily	Admin
TC16	Upload PDF with images only	Upload image-based PDF	Tika returns no usable content, handled gracefully	Admin

TC17	Upload invalid extension	Attempt CSV upload	Error: Unsupported format	Admin

3.6 Token & Session Handling

Test ID	**Scenario**	**Action Description**	**Expected Result**	**Role**
TC18	Logout	User explicitly logs out	JWT removed from frontend/session cleared	User/Admin
TC19	Access with expired token	Access after JWT expiration	403 Forbidden or re-authentication prompted	User/Admin

4. Database Validation

Test ID	**Scenario**	**Database Table**	**Expected Entry/Change**
DB01	User signup	users	New record with username, encoded password, role=USER
DB02	Role change	users	Updated role value (USER → ADMIN / ADMIN → USER)
DB03	Document upload	N/A (file system)	File saved under /uploads and text stored in memory

5. Notes & Observations

- All POST and GET APIs are validated using Swagger and Postman.
- Role-based restrictions work as expected and follow Spring Security configurations.

- Apache Tika handles different file types efficiently, and throws expected exceptions for invalid content.
- LLM API integration is assumed to be successful based on mocked responses or valid external calls.
- Token expiration and manual logout are handled client-side.

6. Conclusion

The test cases validate the core functionality and access control of the application effectively. All major scenarios including file handling, question answering, user management, and authentication are covered. The application shows stable performance with clear behavior for both Admin and Standard Users.

Further test coverage (e.g., for edge cases, scalability, and exception flow) can be added as the system evolves.