Test Plan and Test Cases Document

Automotive Service Center Project

Version: 1.0
Date: November 6, 2025
Project: Automotive Service Center
Prepared By: Test Team
Group Number: 8

Table of Contents

---

1. Test Plan Overview

1.1 Purpose

This document outlines the comprehensive test plan and test cases for the Automotive Service Center microservices backend system. The system consists of multiple microservices that handle user authentication, registration, customer management, service management, mechanic operations, and call center operations.

1.2 Objectives

- Ensure all microservices function correctly as per requirements
- Validate JWT-based authentication and authorization across services
- Verify BCrypt password encryption implementation
- Test role-based access control (RBAC) for each service
- Validate business logic and data integrity
- Ensure proper error handling and exception management
- Verify integration between services

1.3 Scope

This test plan covers:

- Unit Testing: Service layer, controller layer, factory classes
- Integration Testing: Service-to-service communication
- Security Testing: JWT authentication, password encryption, role-based access
- Functional Testing: Business logic validation
- Exception Testing: Error handling and edge cases

1.4 Out of Scope

- Performance/Load Testing
- UI/UX Testing
- Database migration testing
- Network infrastructure testing

---

2. Test Strategy

2.1 Testing Levels

2.1.1 Unit Testing

- Framework: JUnit 5 with Mockito
- Coverage: Service layer
- Approach: Mock external dependencies (repositories, RestClient, etc.)
- Target Coverage: Minimum 80% code coverage

2.1.2 Integration Testing

- Focus: Inter-service communication
- Tools: Postman,Swagger
- Coverage: API endpoints, service interactions

2.1.3 Security Testing

- JWT Token Validation: Token generation, parsing, expiration
- Password Encryption: BCrypt implementation
- Role-Based Access Control: Verify role restrictions

2.2 Testing Approach

- Test-Driven Development (TDD): Tests written before/alongside implementation
- Mocking Strategy: Mock external dependencies to isolate units under test
- Test Data Management: Use test fixtures and builders
- Test Isolation: Each test is independent and can run in any order

2.3 Test Tools and Frameworks

- JUnit 5: Test framework
- Mockito: Mocking framework
- Postman and Swagger: Integration testing
- Maven Surefire Plugin: Test execution

---

3. Test Scope

3.1 Services Under Test

1. Signup Service
    - User registration
    - User factory pattern
    - Password encryption
    - Email notification
2. Login Service
    - User authentication
    - JWT token generation
    - Password validation with BCrypt
    - Role extraction
3. Customer Service
    - Vehicle management
    - Work order creation
    - Service status tracking
4. Service Manager Service
    - Work order management
    - Manager assignment
    - Mechanic assignment
    - Cost updates
5. Mechanic Service
    - Work order retrieval
    - Work order updates
    - Progress tracking
6. Call Centre Service
    - Feedback management

        - Agent assignment
        - Feedback responses
7. Email Service
        - Email sending functionality
        - Template rendering
8. Service Registry (Eureka)
        - Service discovery

3.2 Security Features Under Test

- JWT token generation and validation
- BCrypt password encryption
- Role-based access control (RBAC)
- Unauthorized access prevention

---

4. Test Environment

4.1 Test Environment Setup

- Java Version: 21
- Spring Boot Version: 3.3.4
- Build Tool: Maven
- Database: MySQL (test database)
- Test Framework: JUnit 5, Mockito

4.2 Prerequisites

- Maven installed and configured
- Java 21 JDK
- MySQL database (for integration tests)
- All dependencies resolved

4.3 Test Execution

```
# Run all tests
mvn test

# Run tests for specific service
cd <service-directory>
mvn test

# Run specific test class
mvn test -Dtest=ClassName
```

---

5. Test Cases by Service

5.1 Signup Service

5.1.1 SignUpServiceImpl Test Cases

Test ID / Name	Description	Expected Result	Priority
TC-SU-001	Register a new customer with valid data	User registered successfully, password encrypted, email sent	High
TC-SU-002	Register a service manager with valid data	Service manager registered with experience and department	High
TC-SU-003	Register a mechanic with valid data	Mechanic registered with specialization and hourly rate	High

TC-SU-004	Register an admin user	Admin user registered successfully	High
TC-SU-005	Register a call centre agent	Agent registered successfully	High
TC-SU-006	Attempt to register with existing email	UserAlreadyExistsException thrown	High
TC-SU-007	Attempt to register with existing phone	UserAlreadyExistsException thrown	High
TC-SU-008	Register user when email service fails	User registered successfully, email failure logged	Medium
TC-SU-009	Verify all repository methods called correctly	All repository methods verified	Medium

5.1.2 UserFactory Test Cases

Test ID / Name	Description	Expected Result	Priority
TC-UF-001	Create Customer entity from DTO	Customer object created with correct fields	High
TC-UF-002	Create ServiceManager entity from DTO	ServiceManager created with experience and department	High

TC-UF-003	Create Mechanic entity from DTO	Mechanic created with specialization and availability	High
TC-UF-004	Create AdminUser entity from DTO	AdminUser created successfully	High
TC-UF-005	Create CallCenterAgent entity from DTO	CallCenterAgent created successfully	High
TC-UF-006	Attempt to create user with invalid role	IllegalArgumentException thrown	High

5.1.3 SignUpServiceController Test Cases

Test ID / Name	Description	Expected Result	Priority
TC-SC-001	POST request with valid signup data	HTTP 201 Created, user registered	High
TC-SC-002	POST request with invalid email format	HTTP 400 Bad Request	High
TC-SC-003	POST request with missing fields	HTTP 400 Bad Request	High
TC-SC-004	POST request with existing email	HTTP 409 Conflict	High

5.2 Login Service

5.2.1 LoginServiceImpl Test Cases

Test ID / Name	Description	Expected Result	Priority
TC-LI-001	Login with correct email and password	JWT token generated, LoginResponseDTO returned	High
TC-LI-002	Login with non-existent email	UserNotFoundException thrown	High
TC-LI-003	Login with incorrect password	InvalidPasswordException thrown	High
TC-LI-004	Login with null password	InvalidPasswordException thrown	High
TC-LI-005	Login with empty password	InvalidPasswordException thrown	High
TC-LI-006	Login with different case password	InvalidPasswordException thrown	Medium
TC-LI-007	Login with different roles (CUSTOMER, SERVICE_MANAGER, etc.)	JWT token generated with correct role	High

TC-LI-008	Login with different email case	UserNotFoundException (if case-sensitive)	Medium
TC-LI-009	Verify repository and JWT util calls	All methods verified	Medium
TC-LI-010	Verify BCrypt password matching	PasswordEncoder.matches() called	High

5.3 Customer Service

5.3.1 VehicleServiceImpl Test Cases

Test ID / Name	Description	Expected Result	Priority
TC-VS-001	Add a new vehicle for a customer	Vehicle saved successfully	High
TC-VS-002	Add vehicle for non-existent customer	CustomerDoesNotExistException thrown	High
TC-VS-003	Retrieve all vehicles for a customer	List of vehicles returned	High
TC-VS-004	Get vehicles for non-existent customer	CustomerDoesNotExistException thrown	High

TC-VS-005	Update existing vehicle details	Vehicle updated successfully	High
TC-VS-006	Update non-existent vehicle	VehicleDoesNotExistException thrown	High
TC-VS-007	Delete a vehicle	Vehicle deleted successfully	High
TC-VS-008	Delete non-existent vehicle	VehicleDoesNotExistException thrown	High

5.3.2 WorkOrderServiceImpl Test Cases

Test ID / Name	Description	Expected Result	Priority
TC-WO-001	Create work order for customer	Work order created, email sent	High
TC-WO-002	Create work order for non-existent customer	CustomerDoesNotExistException thrown	High
TC-WO-003	Create work order for non-existent vehicle	VehicleDoesNotExistException thrown	High
TC-WO-004	Create work order for already booked vehicle	VehicleDoesNotExistException thrown	High

TC-WO-005	Get service status for customer	Service status list returned	High
TC-WO-006	Get status for non-existent customer	CustomerDoesNotExistException thrown	High
TC-WO-007	Get status when no service data exists	VehicleDoesNotExistException thrown	Medium

5.4 Service Manager Service

5.4.1 ServiceManagerService Test Cases

Test ID / Name	Description	Expected Result	Priority
TC-SM-001	Retrieve all open work orders	List of open work orders returned	High
TC-SM-002	Retrieve work order by service order ID	Work order details returned	High
TC-SM-003	Retrieve non-existent work order	WorkOrderNotFoundException thrown	High
TC-SM-004	Assign service manager to work order	Manager assigned successfully	High

TC-SM-005	Assign manager to non-existent work order	WorkOrderNotFoundException thrown	High
TC-SM-006	Assign mechanic to work order	Mechanic assigned successfully	High
TC-SM-007	Assign mechanic to non-existent work order	WorkOrderNotFoundException thrown	High
TC-SM-008	Start a work order	Work order status updated to IN_PROGRESS	High
TC-SM-009	Start non-existent work order	WorkOrderNotFoundException thrown	High
TC-SM-010	Complete a work order	Work order status updated to COMPLETED	High
TC-SM-011	Complete non-existent work order	WorkOrderNotFoundException thrown	High
TC-SM-012	Update work order costs	Costs updated successfully	High
TC-SM-013	Update costs for non-existent work order	WorkOrderNotFoundException thrown	High

5.5 Mechanic Service

5.5.1 MechanicServiceImpl Test Cases

Test ID / Name	Description	Expected Result	Priority
TC-ME-001	Get all work orders assigned to mechanic	List of assigned work orders returned	High
TC-ME-002	Get work orders when none assigned	Empty list returned	Medium
TC-ME-003	Retrieve work order by ID	Work order details returned	High
TC-ME-004	Retrieve non-existent work order	WorkOrderNotFoundException thrown	High
TC-ME-005	Start assigned work order	Work order status updated to IN_PROGRESS	High
TC-ME-006	Start non-existent work order	WorkOrderNotFoundException thrown	High
TC-ME-007	Complete assigned work order	Work order status updated to COMPLETED	High
TC-ME-008	Complete non-existent work order	WorkOrderNotFoundException thrown	High

TC-ME-009	Update work order progress	Progress updated successfully	High
TC-ME-010	Update progress for non-existent work order	WorkOrderNotFoundException thrown	High

5.6 Call Centre Service

5.6.1 CallCenterAgentService Test Cases

Test ID / Name	Description	Expected Result	Priority
TC-CC-001	Fetch feedbacks from customer service	Feedbacks retrieved successfully	High
TC-CC-002	Retrieve all feedbacks	List of all feedbacks returned	High
TC-CC-003	Get feedbacks when none exist	Empty list returned	Medium
TC-CC-004	Assign agent to feedback	Agent assigned successfully	High
TC-CC-005	Assign agent to non-existent feedback	FeedbackNotFoundException thrown	High
TC-CC-006	Add response to feedback	Response added successfully	High

TC-CC-007	Respond to non-existent feedback	FeedbackNotFoundException thrown	High
TC-CC-008	Get all feedbacks assigned to agent	List of agent's feedbacks returned	High
TC-CC-009	Get feedbacks when agent has none	Empty list returned	Medium

5.7 Security Test Cases

5.7.1 JWT Authentication Test Cases

Test ID / Name	Description	Expected Result	Priority
TC-JWT-001	Generate token for valid user	Valid JWT token generated with user claims	High
TC-JWT-002	Validate a valid JWT token	Token validated successfully	High
TC-JWT-003	Validate an expired token	Token validation fails	High
TC-JWT-004	Validate a malformed token	Token validation fails	High
TC-JWT-005	Extract user role from token	Correct role extracted	High

TC-JWT-006	Extract user details from token	User ID, email, name extracted correctly	High

5.7.2 Password Encryption Test Cases

Test ID / Name	Description	Expected Result	Priority
TC-PWD-001	Encrypt password using BCrypt	Password encrypted with BCrypt hash	High
TC-PWD-002	Match plain password with encrypted	Password matches successfully	High
TC-PWD-003	Match wrong password with encrypted	Password does not match	High
TC-PWD-004	Encrypt same password twice	Different hashes generated (salt)	High

5.7.3 Role-Based Access Control Test Cases

Test ID / Name	Description	Expected Result	Priority
TC-RBAC-001	Access customer service with CUSTOMER role	Access granted	High

TC-RBAC-002	Access customer service with SERVICE_MANAGER role	Access denied, UnauthorizedException	High
TC-RBAC-003	Access service manager service with correct role	Access granted	High
TC-RBAC-004	Access service manager service with CUSTOMER role	Access denied	High
TC-RBAC-005	Access mechanic service with MECHANIC role	Access granted	High
TC-RBAC-006	Access call centre service with CALL_CENTRE_AGENT role	Access granted	High
TC-RBAC-007	Access protected endpoint without JWT token	Access denied, UnauthorizedException	High
TC-RBAC-008	Access protected endpoint with invalid token	Access denied, UnauthorizedException	High

---

6. Test Execution Summary

6.1 Test Execution Status

Service	Total Test Cases	Passed	Failed	Skipped	Coverage
Signup Service	19	19	0	0	~85%
Login Service	10	10	0	0	~90%
Customer Service	15	15	0	0	~80%
Service Manager Service	13	13	0	0	~85%
Mechanic Service	10	10	0	0	~80%
Call Centre Service	9	9	0	0	~80%
Total	76	76	0	0	~83%

6.2 Test Execution Commands

```
# Run all tests
mvn test

# Run signup service tests
cd signup/signup && mvn test

# Run login service tests
cd login/login && mvn test

# Run customer service tests
cd customer/customer && mvn test
```

```
# Run service manager tests
cd service-manager/service-manager && mvn test

# Run mechanic service tests
cd "mechanic service/mechanic-service" && mvn test

# Run call centre service tests
cd call-centre-service/call-centre-service && mvn test
```

6.3 Test Results Location

Test reports are generated in:

- `target/surefire-reports/` - Test execution reports
- `target/surefire-reports/*.txt` - Text format reports
- `target/surefire-reports/*.xml` - XML format reports (for CI/CD integration)

---

7. Defect Management

7.1 Defect Severity Levels

- Critical: System crash, data loss, security breach
- High: Major functionality broken, cannot proceed
- Medium: Functionality partially broken, workaround available
- Low: Minor issues, cosmetic problems

7.2 Defect Lifecycle

1. New: Defect identified and logged
2. Assigned: Defect assigned to developer
3. In Progress: Developer working on fix
4. Fixed: Fix implemented and ready for testing
5. Verified: Fix verified by tester

6. Closed: Defect resolved and closed

7.3 Defect Tracking

All defects should be logged with:

- Defect ID
- Description
- Steps to reproduce
- Expected vs Actual result
- Severity
- Priority
- Status

---

8. Test Deliverables

8.1 Test Artifacts

1. Test Plan Document (This document)
2. Test Cases Document (This document)
3. Test Code (All test classes in `src/test` directories)
4. Test Execution Reports (Maven Surefire reports)
5. Test Coverage Reports (Code coverage metrics)

8.2 Test Metrics

- Test Coverage: ~83% overall code coverage
- Test Execution Time: ~30-45 seconds for all tests
- Test Pass Rate: 100% (76/76 tests passing)
- Code Quality: All tests follow best practices with proper mocking and assertions

8.3 Test Maintenance

Tests should be updated when:

- New features are added
- Existing features are modified
- Bugs are fixed
- Requirements change

---

9. Test Case Template

9.1 Standard Test Case Format

Test ID: TC-XX-XXX
Test Case Name: [Descriptive name]
Test Description: [What is being tested]
Preconditions: [What must be true before test]
Test Steps:

1. [Step 1]
2. [Step 2]
3. [Step 3]

Test Data: [Input data required]
Expected Result: [What should happen]
Actual Result: [What actually happened]
Status: [Pass/Fail/Blocked]
Priority: [High/Medium/Low]

---

10. Risk Assessment

10.1 Testing Risks

Risk	Impact	Probability	Mitigation
Test environment unavailable	High	Low	Maintain backup test environment
Test data corruption	Medium	Medium	Use test data fixtures, reset between tests
Incomplete test coverage	Medium	Medium	Regular code coverage reviews
Integration issues between services	High	Medium	Integration testing, contract testing
Security vulnerabilities	High	Low	Regular security audits, penetration testing

---

11. Sign-off

11.1 Approval

Role	Name	Signature	Date

Test Lead	Manan Pandya	Manan	06/11/2025
Development Lead	Manan Pandya	Manan	06/11/2025
Project Manager	Manan Pandya	Manan	06/11/2025

---

12. Appendix

12.1 Test Data

User Test Data

- Customer: email: customer@test.com, password: password123
- Service Manager: email: manager@test.com, password: password123
- Mechanic: email: mechanic@test.com, password: password123
- Call Centre Agent: email: agent@test.com, password: password123

Vehicle Test Data

- Vehicle ID: 1
- Customer ID: 1
- Make: Toyota
- Model: Camry
- Year: 2020

Work Order Test Data

- Work Order ID: 1
- Customer ID: 1

- Vehicle ID: 1
- Status: OPEN

12.2 References

- Spring Boot Testing Documentation: https://spring.io/guides/gs/testing-web/
- JUnit 5 User Guide: https://junit.org/junit5/docs/current/user-guide/
- Mockito Documentation: https://javadoc.io/doc/org.mockito/mockito-core/latest/org/mockito/Mockito.html
- BCrypt Documentation: https://docs.spring.io/spring-security/reference/features/authentication/password-storage.html

Document Version History

Version	Date	Author	Changes
1.0	2025-11-06	Hardik Suthar	Initial test plan and test cases document

End of Document