

# ICTAK- Tech Blog-admin

## Test Plan Document

**Project Name:** Web Application Testing – ICTAK - Tech Blog

**URL:** <http://64.227.132.106/>

**Prepared By:** Sayoojya S V

**Role:** SDET

**Date:** Dec 2025

## 1. Introduction

The Article Posting Web Application is a role-based content management platform designed to streamline the creation and publishing of articles. It supports three user roles: User, Trainer, and Admin. Users can write and submit articles but require Admin approval before their content is published, ensuring proper moderation. Trainers, who have higher privileges, can post articles directly without approval. The Admin oversees the entire system by reviewing submissions, approving or rejecting articles, and managing platform activities. This Test Plan aims to validate how each role interacts with the application and ensure that all core functionalities including login, article submission, approval workflows, and navigation operate accurately, consistently, and user-friendly across different scenarios.

## 2. Scope

### In-Scope

- Testing of Admin functionalities, including login, dashboard access, article approval, rejection, editing, and deletion.
- Validation of Admin functionalities.
- Testing of the article lifecycle from creation to approval or publishing.
- Negative testing for invalid inputs, unauthorized actions, and error handling.

## **Out-Scope**

- User and Trainer module
- Performance, load, and stress testing.
- Advanced security testing beyond authentication and authorization.
- Mobile responsiveness and cross-device compatibility testing.

## **3. Objective**

- To validate and ensure correct functioning of all Admin module features.
- To identify, report, and track defects through manual testing and test management tools.
- To automate key Admin workflows using Selenium with a POM-based framework.
- To enable continuous and consistent test execution using Jenkins, Docker, and cloud infrastructure.

## **4. Testing Approach**

- Testing is focused exclusively on the Admin module, as per the defined project scope.
- The application is initially explored to understand Admin workflows, including login, dashboard access, and article management features.
- Manual testing is performed first to gain clarity on system behavior, navigation, and UI flow before automation.
- Test cases are designed to cover positive scenarios, negative scenarios, UI validation, and navigation for Admin functionalities.
- Each test case is executed systematically, and results are documented for effective tracking and reporting.
- Defects are logged with clear steps to reproduce, expected results, and actual outcomes.
- Identified issues are analyzed to assess their impact on Admin operations and overall system stability.

## 5. Test Strategy

The test strategy defines the approach used to ensure the quality of the Admin module of the Article Posting Web Application. Testing follows a sprint-based approach, beginning with manual testing to understand Admin workflows, UI behavior, and navigation. Since testing is executed using Admin access, all validations are performed from the Admin perspective. Once the core Admin functionalities are stabilized, automation testing is introduced to improve test execution efficiency and ensure consistent regression testing.

### 5.1 Types of Testing

- **Functional Testing** : To verify that functionalities related to Admin, User, and Trainer roles such as login, article submission, approval, rejection, and publishing work as expected.
- **UI Testing** : To ensure that the interfaces for Admin are visually consistent, readable, and user-friendly.
- **Navigation Testing** : To check whether page transitions, dropdown, and links work smoothly across different role-based pages.
- **Negative Testing** : To validate application behavior when invalid inputs or unauthorized actions are performed by Admin.
- **Boundary Testing** : To test input limits and edge cases where applicable, ensuring the system handles them correctly.

## 6. Risks & Mitigation

- **Limited direct role access** : Testing execution is primarily done using Admin access, which may limit direct interaction with User and Trainer roles. To mitigate this, User and Trainer workflows are validated through Admin-side approval and monitoring processes.
- **Application instability** : Occasional slowness or downtime may impact testing activities. Testing is resumed once the application stabilizes, and all such issues are documented clearly.
- **Changes during sprints** : Requirement or workflow changes may affect existing test cases. Test cases and documents are updated promptly whenever changes are identified.

- **Time limitations** : Short sprint durations may limit full test execution. Critical workflows for Admin, User, and Trainer are prioritized to ensure maximum coverage.

## **7. Roles and Responsibilities**

- Verifies Admin login workflows and ensures role-based access works correctly.
- Tests all role-related functionalities and confirms each page responds as expected.
- Validates article submission, approval, rejection, and publishing workflows.
- Ensures smooth navigation across Admin pages.
- Checks form validations and confirms appropriate error or success messages are displayed.
- Identifies defects during testing and reports them clearly with supporting details.
- Performs retesting after fixes to confirm issues are resolved.

## **8. Entry/Exit Criteria**

### **Entry Criteria :**

- The application is accessible and stable for Admin module testing.
- Valid Admin login credentials are available and working.
- The test environment (system, browser, tools) is properly configured.
- Admin workflows such as login, dashboard access, and article management are clearly understood.
- Test cases for the Admin module are prepared and reviewed prior to execution.

### **Exit Criteria :**

- All planned Admin module test cases are executed successfully.
- Critical and major defects identified during testing are logged and tracked.
- Retesting is completed for defects marked as fixed.
- Test execution results and defect details are clearly documented.
- All testing deliverables are completed and ready for final review.

## 9. Test Environment

### Software

- Operating System: Windows 10 / Windows 11
- Testing Tools: Selenium WebDriver, Postman
- Programming Language: Java
- Build Tool: Maven
- Test Framework: TestNG
- Version Control: GitHub

### Hardware

- System Type: Desktop / Laptop
- Processor: Minimum Intel i3 or equivalent
- RAM: Minimum 8 GB
- Storage: Minimum 100 GB free space
- Internet: Stable broadband connection

### Browser

- Google Chrome (Primary browser for testing)
- Microsoft Edge (Secondary browser for compatibility testing)

## 10. Test Deliverables

- **Requirement Summary** outlining key features and Admin workflows of the application.
- **Test Plan document** defining the testing scope, approach, environment, risks, and responsibilities.
- **Test Case Document** containing detailed test cases with steps, expected results, and priorities.

- **Test Execution Report** summarizing executed test cases, pass/fail status, and observations.(Test-Rail)
- **Bug Report** documenting identified defects with severity, priority, and screenshots.
- **Automation Framework** implemented using Selenium with Page Object Model.
- **Automated Test Scripts** covering critical Admin functionalities.
- **Automation Test Reports** (Extent / Allure) with execution results and screenshots.
- **GitHub Repository** containing all test documents, automation code, and README instructions.
- **Jira-Zephyr** test case report