ICTAK-Institution-Portal

Test Plan Document

Project Name: Web Application Testing- ICTAK-Institution Portal
URL: *https://betadev.ictkerala.org/app/institution/signin*
Prepared By: Group 2(Sayoojya, Samitha ,Hasna, Karnan)
Date: 16-01-2026

1. Introduction

The Institution Portal is designed to engage partner institutions by providing access to program details, facilitating MoU execution, and enabling the submission of nominations or registrations for credit-based programs. This End-to-End Test Plan outlines the approach to validate the complete workflow of the portal, ensuring all functionalities operate seamlessly, securely, and in alignment with business requirements before production release.

2. Scope

In Scope

- Institution Login
- Dashboard access and navigation
- Viewing and searching program details
- Institution Request Module
- MoU initiation, upload, submission, and status tracking
- Bulk Nomination
- Data validation and error handling
- Reports and nomination history viewing
- Payment Module
- Role-based access and basic security and performance checks
- Cross-browser and end-to-end workflow validation

Out-of-Scope

- Internal admin portal functionality (if managed separately)
- Third-party payment gateway internal logic
- Infrastructure, server, and database-level testing
- Mobile application testing

3. Objective

- Validate all functional requirements work as expected
- Ensure role-based access control between Institution users and Admin users
- All workflows are complete and consistent
- Data flows correctly from front-end to back-end
- Verify security, performance, usability and payment reliability.
- Identify defects early and ensure a stable production-ready application.

4. Testing Approach

4.1 Overview

The testing approach for the Institution Portal focuses on validating functional correctness, usability, security, and data integrity from an end-user perspective. Since formal requirement documentation is not available, testing is performed based on business understanding, application behavior, and standard industry practices.

Testing Methodology

- **Manual Testing** for exploratory, usability, and first-time validation.
- **Automation Testing** to validate repetitive and critical scenarios, ensuring faster feedback and consistent test execution for stable application modules.
- **Regression Testing** after each build or defect fix.

Test Levels Covered

- **End-to-End (E2E) Testing**
- **Integration Testing**
- **User Acceptance Perspective Testing**

Functional Testing

Functional testing will validate the following modules:

- Login & Session Management
- Institution Requests
- MoU Viewing and Download
- Nomination
- Payment Processing
- Nomination History & Search
- FAQ & Help Pages

UI & Usability Testing

- Page alignment and layout consistency
- Loading indicators vs empty states
- Error and success message clarity
- Pagination and navigation behavior
- Sorting and ordering

5. Test Environment

- **Hardware:** System with minimum 8GB RAM.
- **Software:**
    - TestRail for Test Management
    - Selenium for Test Automation
    - JIRA for project management and defect tracking
    - GitHub for Version Control.
- **Browsers:** Google Chrome, Firefox, Edge

6. Entry/ Exit Criteria

Entry Criteria

- Test environment is stable and accessible
- Functional requirements are finalized and approved
- Test cases are reviewed and approved in TestRail
- Required test data and credentials are available

Exit Criteria

- All critical and high-priority test cases are executed
- No open critical or blocker defects
- High-priority defects are either fixed or accepted with mitigation
- Test summary report is prepared and shared
- Application is deemed stable for production deployment

7. Risks and Mitigation

- Incomplete or unclear requirements: Mitigated through early review sessions and continuous clarification with stakeholders.
- Bulk upload failures or data issues: Mitigated by testing with multiple file formats, sizes, and valid/invalid data combinations.
- Payment-related issues: Mitigated by validating portal-side payment flow, error handling, and transaction status updates.
- Test environment instability: Mitigated by allocating buffer time for execution and scheduling retest windows.
- Limited or unrealistic test data: Mitigated by preparing representative dummy data in advance.

8. Test Deliverables

- Test Plan document
- Test cases maintained in TestRail
- Test execution reports
- Defect reports with evidence (screenshots)
- Regression test results