Test Plan

Khimasia Krish - 21CS10037
Tanishq Prasad - 21CS30054
Sourodeep Datta - 21CS10064

1 TEST PLAN IDENTIFIER

This is Test Plan 1 for the first version of the Hall Management System project.

2 References

1. Software Requirements Specification for this project

3 Introduction

The purpose of this test plan is to lay out a methodical framework for testing the various functionalities described in the SRS document submitted earlier. This is needed to ensure that the various components of this web application are working as expected in a wide variety of situations. This helps in delivering a smooth, glitch-free user experience to the students, wardens, etc.

4 TEST ITEMS

1. Employee management system for the Hall Management Centre that takes care of employee registration, salary payments, leave, etc.
2. Financial accounting for various halls
3. Student management system that takes care of room allotment.
4. Fees and dues management
5. The complaint resolution system with its associated user interfaces that differ for different end users (such as students, wardens, hall clerks, etc.).
6. Temporary Employees' Salary Management
7. Petty expense management for the Hall Management Centre

5 SOFTWARE RISK ISSUES

1. Inconsistent database entries. This can occur if the CRUD (Create-Read-Update-Delete) operations performed on the database are not synchronized properly with each other (for example, race conditions). These inconsistencies can result in fatal errors later on if the web app is unable to determine valid records for reading or updating.

2. Incorrect user management and privilege separation. Different classes of users have different roles to play in the web app, and certain classes are not privy to certain information and abilities. They should not be allowed to perform certain actions. For example, students should not be allowed to submit action-taken reports for complaints filed by other students. That is a task that should be performed only by the wardens. If students are able to log in as wardens, this could result in a catastrophic lack of accountability.

3. The web application should not crash and be responsive under a huge load. Since IIT KGP is a huge institute with thousands of students, the app must be capable of handling thousands of simultaneous requests. Therefore, the final configuration should be tested against this possibility to ensure the servers don't deny user requests because of high loads.

6 FEATURES TO BE TESTED

These are the features that are to be tested from the user's perspective. This primarily consists of user-facing features that are handled by a graphical web page that allows privileged access to the internal server operations.

Procedure: These features will be tested by following a sequence of instructions for navigating the web pages associated with the application, followed by a verification of the final internal state of the server-side application.

1. Employee Management (**Priority**: H)

   (a) For the HMC, this refers to the management of regular employees and their salaries.

   (b) For the halls, this refers to the management of temporary day-to-day employees whose salaries and attendance need to be accounted for.

   (c) Approving and keeping track of leaves taken by the hall employees.

2. Financial Accounting For The Halls (**Priority**: H)

   (a) HMC Chairman should be able to allocate grants to different halls.

   (b) Wardens should be able to send expense reports to the HMC Chairman.

   (c) Wardens should be able to view and print a full account of the expenses and payments made and send this for an audit if required.

   (d) HMC Chairman should be able to earmark money for various halls based on the grants received by the HMC.

3. Student Management System (**Priority**: M)

   (a) Room allotment.

4. Fees And Dues Management (**Priority**: H)

   (a) Students should be able to view expenses incurred from their end and settle dues.

   (b) Mess managers must be able to enter the money owed by students individually.

   (c) Wardens must be able to settle mess manager accounts based on data given to them by the managers.

   (d) Wardens should be able to set rent amounts, charges for different amenities, etc.

5. Complaint Resolution System (**Priority**: M)

   (a) Students should be able to file complaints and view complaint history.

   (b) Wardens should be able to view and close complaints with Action Taken Reports along with an eagle-eye history of all the complaints that have been filed.

   (c) Hall clerks should be able to view complaints against them while keeping the complainant anonymous.

   (d) Action taken reports should be publicly accessible for everyone to see

6. Petty Expense Management (**Priority**: L)
   The HMC incurs petty expenses, and the application must be capable of recording those expenses and displaying a report of those expenses.

7. Production Of Various Reports (**Priority**: M)
   (a) Financial reports as requested by the wardens/HMC Chairman
   (b) Petty expense reports
   (c) Room vacancy and occupancy reports for wardens along with student information
   (d) Employee management reports
   (e) Complaint reports

7 FEATURES NOT TO BE TESTED

There are no such features since the web application won't have future revisions. Therefore, no hidden disabled features will be present. Since this is a one-off project with no further maintenance, all features need to be tested.

8 APPROACH (STRATEGY)

1. General Interface Testing
   (a) The tester will open the web application on various browsers to test for browser compatibility issues.
   (b) They will click on all the different links on the web page to ensure no part of the user interface contains broken links.
   (c) They will try accessing web pages that should not be accessible to them to verify if the authentication system is bug-free.
2. Testing Of Various Specific Subsystems
   (a) The tester will open up the relevant web pages for the subsystem considered and will follow the instructions provided in the test cases. These instructions will include things like which button to click, etc, followed by the expected system response.
   (b) If there's a mismatch between the actual system response and expectations, a bug report will be filed, following which the developers will try to isolate the cause and fix it.

3. Testing Client Compatibility

   (a) Apart from testing browser compatibility, the tester should also use various devices to ensure the application works smoothly across operating systems, and hardware platforms.

   (b) They should test it on platforms such as Android devices, Apple devices, generic desktops, and laptops.

   (c) Internet bandwidth tests need to be performed to see if the web app is functional in low-bandwidth situations since it might be possible that poor network conditions prevail and the application must work even during such situations.

   (d) The tester should also check if the web pages are rendered correctly for different configurations involving fonts, browser settings, JavaScript permissions, cookie permissions, etc.

4. What metrics will the tester collect?

   (a) Web page loading times

   (b) Response when under load

   (c) Errors involving bugs in the system

   (d) Differences across various client configurations (such as devices)

   (e) Potential user interface complications such as convoluted menus and submenus.

9 ITEM PASS FAIL CRITERIA

1. All unit tests involving specific isolated subsystems need to be successfully working. No test can fail since any test failure implies that the subsystem is not properly accessible to the end user. A release version of this application cannot tolerate such issues.

2. All client configurations must be tested, and if the application is not working on niche platforms (such as the older versions of Internet Explorer), they should be clearly documented. The test will be considered successful if the application runs under reasonably modern systems. It **DOES NOT** have to work on older systems that are no longer in wide use. This information will be relayed to the end-user, who will be asked to modify their configuration.

10 SUSPENSION CRITERIA AND RESUMPTION REQUIREMENTS

The tests associated with a subsystem should be suspended in only one case, i.e. when the main web page associated with the specific subsystem does not load in a client. This means other test cases related to that subsystem cannot even proceed. This is the only case where suspension should occur. The tests need not be suspended otherwise, since the isolation of subsystems is a core principle. Test failures in specific subsystems can be used to fix errors pertaining to only those subsystems, thus guaranteeing that this failure is completely contained within a specific unit. This also tells us how various components interact with each other, and suspending the tests means we won't be able to gather sufficient information. Resumption occurs once we fix the errors associated with the test failures that prevent the main pages from loading.

11 TEST DELIVERABLES

1. Test plan
2. Test cases
3. Bug reports
4. Error logs

12 SCHEDULE

The testing will occur simultaneously with the development. The final date is 9 April 2023.