Software Test Plan Document
Using IEEE 829-2208 Standard

<Web application for Abuse report>

Course CS509 – Design of Software Systems

Group2:
Chengjiao Yang
Mohammed Ayub
Qiukun Lin
Rui Jin
Rundong Yu
Wenxin Zhao
Yichen Lin

Contents

This page intentionally left blank

1 Introduction

The following document describes the test planning of our Abuse Report Web Application in detail; the hardware, software and personnel required, the functionalities to be tested, the approach, schedule to complete the testing and the risks involved with the plan.

1.1 Objectives

We propose to carry out the only some phases of the testing due to strict time constraints. Unit testing will required be carried out on all of the required functionality and if time permits we will also do the System and Integration testing. We plan to finish the testing within two weeks after the design of the system.

1.2 Testing Strategy

This web application is built from the scratch and does not have any previous versions. Hence all the functionalities have to be tested and recorded for future use by the team. Section 5 of this document gives more clear view of the strategy used for testing our system.

1.3 Scope

Since our product will be built in an iterative manner, later updates to the system will require the test plans to be developed for each of the phase. If there are new versions of the system released we will comply with the Change Management and the Configuration Management requirements

1.4 Reference Material

- Problem Statement
- Software Requirement Specification
- Software Design Document
- HIPAA guidelines
- Section 508 Standards
- CMR 4.0 Standards
- IEEE 1008, ANSI/IEEE 1008 - 1987 (Software Unit Testing).
- IEEE Standard for Software Test Documentation (Std 829-1998)

2 Test Items

The following items will be tested or used as references for developing the test plan:

- Requirements specification
- Design specification
- Features (availability, response time)

2.1 Program Modules

According to the required business functionalities of the system, we divide the web application to several independent modules:

1. Log in module – which include verifying the user information, login functionality and capturing unauthorized access.

2. Abuse Report module – which includes creation of report, validate the input information in the forma and required functions like viewing, editing, delete, saving, sending and displaying status of reports.

3. Investigation module – which includes creating new investigation files, track public log number, adding disposition letter, decision letter, response, viewing investigation and editing investigation functions to a particular report.

4. User manager module including add user (staff, supervisor, Human Right Chairman), editing and deleting functions pertaining to the user.

5. Client manager module which includes adding, editing and deleting client functions.

6. Appeal module which including add appeal and track appeal functions.

7. Authority module which includes setting permission levels to each user, displaying different menu options for different kind of users and access control for web URL functions.

8. User Interface module which includes navigation for the web pages, the CSS page styling functions.

9. Document processing module which includes generate official format of PDF, printing, scanning and attaching documents.

10. Audit trail module which includes recording user actions, error messages and application accesses to the system.

3 Features to Be Tested

The following features are to be tested:

1. Functionality: All the ten required functions available in the problem statement need to be tested.
   a) Login Functionality
   b) Writing Abuse Report
   c) Submitting Abuse Report
   d) Modifying Abuse Report
   e) Viewing Abuse Report
   f) Sending Notification Functionality
   g) Printing Functionality
   h) Scanning Functionality
   i) Track Appeal Functionality
   j) Audit Trail Functionality
   k) Displaying the Calendar Functionality for Reports
2. Content: Encrypted information content on display pages should be tested
3. Navigation: Links for navigating to different parts of the system should be tested which includes showing appropriate error pages if any error occurs during navigation.
4. Legal: The Section 508, HIPAA, CMR 4.0 standards should be tested during the verification and validation of system.
5. Security:
   a) The HTTPS functionality of the system should be tested.
   b) Session handling feature for timeout should be tested.

c) Security Access Control for different users to the system should be tested.

4 Features Might Not Be Tested

The additional functionalities, due to limited time constraint might not be tested. This includes the following:

a) Track Appeal Functionality

b) Audit Trail Functionality

c) Displaying the Calendar Functionality for Reports

d) Virtual Investigation File Functionality

5 Approach

We plan to approach the testing based on the priority levels of each test case. Each of the test cases will be prioritized as High, Medium and Low and then scheduled it according to the number of test cases in each. But, if there tend to be a large number of low priority test cases which can be executed with minimum resources those will be tackled first.

5.1 Component/Unit Testing

This test plan outlines the standard tests that should be applied to the group of program files (HTML, JSP, SQL code, etc.) that are used to build each Web page i.e. the tests listed in this document should be used as a template when building the test plan for each Web page (A.K.A. Module).This testing phase will use a number of testing techniques: code reviews and inspections, white-box testing, "buddy" testing etc. to ensure that the code matches the required program specifications. The decision as to which technique(s) to use for any given unit of code will reside with the team leader responsible for signing-off on the Module.

We will start off with writing JUnit test cases for each of the components, to check the correctness and completeness of the logic of each of the required functionality, if any error encountered it will be recorded and reported and all test results will be documented in the test case template.

5.2 Integration Testing

Integration or Incremental testing aims to reduce the need for scaffolding code by using the actual code modules as they are developed to provide the scaffolding for testing. Integration or Incremental test provides a means of carrying out unit tests but at the same time it tests the integration of the system across module boundaries.

We will use the bottom-up approach to test the sub systems of our application. And below is the advantages:

1. Helpful if errors are likely deep down in the dependency structure (e.g. in hardware specific code).
2. Test conditions are easier to create.
3. Observation of test results is reasonably easy.
4. Reduced effort in creating stub modules.

6 Pass/Fail Criteria

The entrance criteria for testing each required functionality should be met before we move onto testing the next required functionality. If the required test case output has been obtained successfully after performing several tests on each functionality, the test case is set to Pass; otherwise it is set to Fail and more in detail analysis needs to be performed.

7 Testing Process

The following represents the overall flow of the testing process:

1. Identify the requirements to be tested. All test cases shall be derived using the current Requirement Specification and System Design.

2. Identify which particular test(s) will be used to test each module.

3. Review the test data and test cases to ensure that the unit has been thoroughly verified and that the test data and test cases are adequate to verify proper operation of the unit.

4. Identify the expected results for each test.

5. Document the test case configuration, test data, and expected results.

6. Perform the test(s).

7. Successful unit testing is required before the unit is eligible for component integration testing.

8. Unsuccessful testing requires a Bug Report Form to be generated. This document shall describe the test case, the problem encountered, its possible cause, and the sequence of events that led to the problem. It shall be used as a basis for later technical analysis.

9. Test documents and reports shall be submitted. Any specifications to be reviewed, revised, or updated shall be handled immediately.

7.1 Test Deliverables

The following documents will be submitted as part of the testing phase:

1. Master test plan (MTP - this document)

2. Individual test cases for the required and additional functionality.

3. Test log for each test cases.

4. Automated test scripts and code supporting test data.

All the above documents will delivered using Microsoft Word or Text files in either a zip file. Automated scripts and code will be in java files and zipped into one file.

7.2 Responsibilities

The following people will be responsible for:

1. Chengjiao Yang (Developer) – Perform developer level testing.

2. Mohammed Ayub (PM, Database Designer) – Database design testing, Operational issues.

3. Qiukun Lin (Tester) – Unit and Integration Testing

4. Rui Jin (Developer) – Perform developer level testing.

5. Rundong Yu (Tester) – Unit and Integration Testing

6. Wenxin Zhao (Database Designer) – Database design testing.

7. Yichen Lin (Developer) – Testing final Design of User Interface.

7.3 Schedule

The following tentative schedule will hopefully be meet:
1. Test design (this document) is expected to be completed by the end of April 3rd.
2. Test execution is expected to last no more than two weeks which will include testing of all requirements as mentioned above and to start immediately after the test Cases have been designed and the Web application has been hosted in a temporary server.
3. Producing the Test Case report and the test log is expected to be completed within two weeks.

8 Environmental Requirements

8.1 Hardware

We decided to test the system on our existing set of desktop and laptop machines, which currently consists of the following machine specifications:

1. Desktop Intel Core2 Duo 2.16Ghz , 512 MB RAM, 80 GB HD, 17" Color Screen (default 1024 x 768 – 16 bit color), external stereo speakers and 56.6kps Modem or 100MB Ethernet Internet connection typically running Windows 7 Ultimate 32bit version.
2. Laptop – Intel Pentium Dual , 1.86 GHz, 2GB RAM, 80 GB HD, 15" Color Screen (default 800 x 600 – 16 bit color), built in stereo speakers and 56.6kps Modem or 10MB Ethernet Internet connection typically running Windows 7 Professional 64 bit version.

Note: All the systems have access to the local printer setup and access to modem (if any).

8.2 Software

The following software are required to conduct the testing:
1. MyEclipse 10
2. Jdk 1.7
3. MySQL 5.6
4. JUnit – Test cases for doing unit testing.
5. LoadRunner- To perform the additional functional test if required by developer/senior tester.

8.3 Security

This will be performed in local network to check for https access and session handling while connecting to the internet and checking for unauthorized accesses to the system.

8.4 Publications

1. The test cases developed for each of the required functionality.
2. The software requirement document.
3. The software design document to get detailed view of the system.

8.5 Risks and Assumptions

1. If there are large number of defects then if becomes functionally impossible to test all of the test

cases considering the amount of testing resources.

2. If Web Server goes down, this will slow down the testing phase and nothing can be done until it is up- many need to reduce the number of test cases.

3. Not enough time to complete all test cases. If time cannot be extended, additional requirement test cases will be skipped, starting with the lowest priority.

9. Test Cases:

9.1 Login Test Case

Test Case Identifier	Login Use Case
Test-case name	User login
Test location	http://host:8080/jsp/test-cases/testlogin.jsp
Feature to be tested	Different users with respective accounts log into the system
Feature pass/failure criteria	The test passes if the correct account could be verified and the application automatically display the main page of named user and the false accounts which can't be verified are marked as fail and after three attempts the account is locked.
Means of control	The *login()* method is called via a test driver *login*
Data	1. Information of username and password are read from an input file http://host:8080/jsp/test-cases/test1. 2. If debug is set to TRUE, then the test case will output the system messages "Login Successful'' for each USERNAME, where USERNAME is the username of current user who wants to log into the system 3. If debug is set to FALSE, then the test case will output the system message "Login Unsuccessful" or "Please Enter correct Username or Password".
Test procedure	1. Click 'Login' button, the system display a login page to enter user name and password. 2. Enter username and password, the system provides the function to type in words in username and password fields. Password must show encrypted symbols during typing. The username must be 5-15 digit Alphabet or Alphabet combined with Arabic number. The password

	must be 8-15 digits alphanumeric characters with combination of Uppercase, lowercase letters and special characters. 3. Click 'login' button in the login page. If enter correctly, the systems verify and automatically transfer to the main page of asserted user. If wrong, system display error message and goes back to main login page. If 3 times failure, the account is locked.
Special requirement	

9.2 Write Report Test Case

Test Case Identifier	Write Report Use Case
Test-case name	Staff writes a new report
Test location	http://host:8080/jsp/test-cases/testwritereport.jsp
Feature to be tested	User-staff creates a new abuse report
Feature pass/failure criteria	The test passes if all the mandatory fields are filled in the report and submitted. It fails if the mandatory fields are empty and user cannot submit the form.
Means of control	The *writereport()* method is called via a test driver *writereport*
Data	1. Abuse report information are read from an input file http://host:8080/jsp/test-cases/test2. 2. If debug is set to TRUE, then the test case will output the system messages "Report Successfully Submitted" 3. If debug is set to FALSE, then the test case will output the system messages "Please fill the required fields".
Test procedure	1. On staff main page, staff click 'new report' button, the system display an empty form of abuse report according to the requirement of DDS. 2. Fill the details of abuse report, the system provides the ability to edit words filled in the report form. 3. Staff click' save' button at the bottom of form, the

	system save all the information the staff had already typed in. 4. Staff click' return' button at the bottom of form, the system automatically goes back to the staff main page 5. On staff main page, staff click' report list', the system display all the reports created by the staff. 6. Staff click the 'edit ' button at the end of the report which staff want to edit and the one which has not been yet submitted, the system display the report form which staff selected with all the information that already saved before. The system provides the ability to edit words filled in the report form. 7. Without filling all the required area of report form and Click' submit 'button at the bottom of form, the system display an message remind staff there are contents missing in the abuse report and goes back to the page with everything staff had already typed in. 8. Filled all the required area of report form and Click' submit ' button at the bottom of form, the systems display a success message to remind the staff the report had already been submitted to the supervisor. 9. Staff click' return' button at the bottom of form, the system automatically goes back to the staff main page
Special requirement	Staff is successfully logged into the system

9.3 Modify Report Test Case

Test Case Identifier	Modify Report Use Case
Test-case name	Supervisor modify the report
Test location	http://host:8080/jsp/test-cases/testmodify.jsp
Feature to be tested	Supervisor modify reports submitted by staff
Feature pass/failure criteria	The test passes if the supervisor could view and modify the abuse report created by the staff.

Means of control	The *Modify()* method is called via a test driver *modify*
Data	1. Information of what need to be modified are read from an input file http://host:8080/jsp/test-cases/test3. 2. If debug is set to TRUE, then the test case will output the system messages "Modify report is successful" 3. If debug is set to FALSE, then the test case will output the system messages "Modify report is failed"
Test procedure	1. On supervisor main page, supervisor click' unfinished report list', The system display all the reports submitted by staff 2. Supervisor click the 'modify' button at the end of the report which staff want to edit, the system display the report form submitted by the staff. The system provides the ability to edit words filled in the report form 3. Supervisor click' save' button at the bottom of form, the system save all the information the supervisor had already modified. 4. Without filling all the required area of report form and Click' finish ' button, the systems display an message remind supervisor there are contents missing in the abuse report and go back to the report form. 5. Filled all the required area of report form and Click' finish' button, the systems display an success message to remind the abuse report is submitted successfully. The system remove this report from 'unfinished report list', and automatically transfer this report to 'finished report list'. 6. Supervisor click' return' button at the bottom of form, the system automatically goes back to the supervisor main page
Special requirement	Supervisor is successfully logged into the system

9.4 View Report Test Case

Test Case Identifier	View Report Use Case
Test-case name	Supervisor view report
Test location	http://host:8080/jsp/test-cases/testview.jsp
Feature to be tested	Supervisor view abuse report
Feature pass/failure criteria	The test passes if the named report could be opened and viewed by Staff/Supervisor/ HRC Chairman.
Means of control	The *view()* method is called via a test driver *view*
Data	1. The public log number of which report supervisor want to view is read from an input file http://host:8080/jsp/test-cases/test5. 2. If debug is set to TRUE, then the test case will output the system messages "View report is successful" 3. If debug is set to FALSE, then the test case will output the system messages "View report is failed."
Test procedure	1. On supervisor main page, supervisor click 'finished report list', the system display all the reports modified by supervisor. Or supervisor click' unfinished report list', the system display all the reports submitted by staff. 2. Supervisor click the report he wants to view or search named report by public log number. The system provides browse function by input public log number. 3. Supervisor click the 'view' button at the end of the report name, the system display the chosen file either in PDF or word format. The report being viewed cannot be edited. 4. Supervisor click' return' button at the bottom of form, the system automatically goes back to report name page.
Special requirement	Supervisor is successfully logged into the system

9.5 Print Report Test Case

Test Case Identifier	Print Report Use Case
Test-case name	Supervisor send the final abuse report
Test location	http://host:8080/jsp/test-cases/testsend.jsp
Feature to be tested	Supervisor view and print the final abuse report
Feature pass/failure criteria	The test passes if supervisor could view and print final abuse report
Means of control	The *print()* method is called via a test driver *print*
Data	1. The public log number of which report supervisor want to send is read from an input file http://host:8080/jsp/test-cases/test4. 2. If debug is set to TRUE, then the test case will output the system messages "Send report is successful 3. If debug is set to FALSE, then the test case will output the system messages "Send report is failed.
Test procedure	1. On supervisor main page, supervisor click 'finished report list', the system display all the reports modified by supervisor. 2. Supervisor click the report he wants to send or search named report by public log number. The systems provide browse function by input public log number. 3. Supervisor click the 'view' button at the end of the report name, the system display the chosen file either in PDF or word format. The report being viewed could not be edited. 4. Supervisor click' return' button at the bottom of form, the system automatically goes back to report name page. 5. Supervisor click' print' button at the bottom of form, the system display the chosen file in PDF format. Right click mouse, click print. The report is sent to printer. 6. Supervisor click' return' button at the bottom of form, the system automatically goes back to report name page.

	7. The supervisor got the copy of abuse report, whether fax or mail to the DPPC.
Special requirement	The user system should be connected to a local Printer capable of printing PDF's.

9.6 Track Report Test Case

Test Case Identifier	Track Report Use Case
Test case name	Supervisor track report
Test location	http://host:8080/jsp/testtrackreport.jsp
Feature to be tested	Supervisor track report status including view investigation files of named report and also purges documents which are passed retention criteria.
Feature pass/failure criteria	Test pass if the named report and associated files could be tracked and purged.
Means of control	The *track ()* method is called via a test driver *track.*
Data	1. The public log number of which report supervisor want to track is read from an input file http://host:8080/jsp/test-cases/test7. 2. If debug is set to TRUE, then the test case will output the system messages "Track report is successful". 3. If debug is set to FALSE, then the test case will output the system messages "Track report is failed".
Test procedure	1. On supervisor main page, supervisor click 'Track report' button, the system display all the reports in the finished report list. 2. Supervisor click the report he wants to track or search named report by public log number. The system provides browse function by input public log number. 3. Supervisor click the 'track' button at the end of the report name, the system display a list of all the files assorted with the abuse report including initial report created by the staff, modified report by supervisor, disposition letter and decision letter from DPPC, appeal from HRC(last 3 may

	not be there at the time) 4. Click view button at the end of the file. The system provides the ability to view the file in either pdf or word format. 5. Click purge button at the end of the file. The system provides the ability to purge named document from the list. 6. Supervisor click' return' button at the bottom of page, the system automatically goes back to report name page.
Special requirement	Supervisor is successfully logged into the system

9.7 Attach Letter Test Case

Test Case Identifier	Attach Letter Use Case
Test case name	Supervisor scan files
Test location	http://host:8080/jsp/testscanfiles.jsp
Feature to be tested	Supervisor attached files to the named report
Feature pass/failure criteria	Test pass if the files (disposition letter, decision letter, and appeal from HRC) could be attached to named abuse report.
Means of control	The *attach ()* method is called via a test driver *attach.*
Data	1. The public log number of which report supervisor want to attach files is and the files which supervisor want to attach are read from an input file http://host:8080/jsp/test-cases/test8. 2. If debug is set to TRUE, then the test case will output the system messages "Scan files is successful". 3. If debug is set to FALSE, then the test case will output the system messages "Scan files is failed".
Test procedure	1. On supervisor main page, supervisor click 'View files' button, the system display all the reports in the finished report list. 2. Supervisor click the report he wants to attach the files to or search named report by public log number. The system provides browse function by

	input public log number. 3. Supervisor click the 'attach' button at the end of the report name, the system display a new page with three column which are disposition letter and decision letter from DPPC, appeal from HRC, there are 'browse' and 'upload' button under each column. 4. Supervisor choose the fields which qualify the file he wants to attach and click 'browse' button, supervisor choose the address of the already scanned document. 5. Supervisor click upload button, the system stored the files into the database and saved the file into the track list of the named abuse report where supervisor could track the file. 6. Supervisor click 'return' button at the end of the page, the system go back to attach main page.
Special requirement	1. Supervisor is successfully logged into the system. 2. The files are already scanned and are stored on the local drive.

9.8 Track Report Test Case

Test Case Identifier	Track Report Use Case
Test case name	Supervisor track report
Test location	http://host:8080/jsp/testtrackappeal.jsp
Feature to be tested	Supervisor track appeal of a named report
Feature pass/failure criteria	Test pass if the appeal of a named report could be viewed
Means of control	The *trackappeal ()* method is called via a test driver *trackappeal.*
Data	1. The public log number of which report supervisor want to track is read from an input file http://host:8080/jsp/test-cases/test9. 2. If debug is set to TRUE, then the test case will output the system messages "Track appeal is successful".

	3. If debug is set to FALSE, then the test case will output the system messages "Track appeal is failed".
Test procedure	1. On supervisor main page, supervisor click 'Track report' button, the system display all the reports in the finished report list. 2. Supervisor click the report he wants to track or search named report by public log number. The system provides browse function by input public log number. 3. Supervisor click the 'track' button at the end of the report name, the system display a list of all the files assorted with the abuse report including initial report created by the staff, modified report by supervisor, disposition letter and decision letter from DPPC, appeal from HRC(last 3 may not be there at the time) 4. Click view button at the end of the file. The system provides the ability to view the file in either pdf or word format. 5. Supervisor click' return' button at the bottom of page, the system automatically goes back to report name page.
Special requirement	Supervisor is successfully logged into the system.

9.9 Audit Trail Test Case

Test Case Identifier	Audit Trail Use Case
Test-case name	Audit trail of accesses
Test location	http://host:8080/jsp/test-cases/testaudittrail.jsp
Feature to be tested	Audit all the access to the system
Feature pass/failure criteria	The test passes if all the access to the system could be counted and audited
Means of control	The *audittrail()* method is called via a test driver *audittrail*
Data	1. Information of command is read from an input file http://host:8080/jsp/test-cases/test10.

	2. If debug is set to TRUE, then the test case will output the system messages “Audit trail” is successful 3. If debug is set to FALSE, then the test case will output the system messages “Audit trail” is failed.
Test procedure	1. Click ‘Audit trail’ button, the system display a list of record of all the access to the system with the information of date accessed, modified and username.
Special requirement	Administrator is successfully logged into the system.

9.11 Calendar Display Test Case

Test Case Identifier	Calendar Display Use Case
Test-case name	Calendar display
Test location	http://host:8080/jsp/test-cases/calendardisplay.jsp
Feature to be tested	Calendar display of all the files associated with a named file
Feature pass/failure criteria	The test passes if there could be a calendar display of all files associated with a named file
Means of control	The *calendardisplay()* method is called via a test driver *calendardisplay*.
Data	1. Public log number of the report that want to be calendar display is read from an input file http://host:8080/jsp/test-cases/test10. 2. If debug is set to TRUE, then the test case will output the system messages “Calendar Display” is successful 3. If debug is set to FALSE, then the test case will output the system messages “Calendar Display” is failed.
Test procedure	1. Click ‘Calendar display’ button, the system display a calendar of the present year. 2. It will also contain the dates marked with reports that have been submitted for investigation or waiting for appeal from Human Rights Committee and tracing is possible through this calendar.

	3. More details of the report can be viewed when we click particular report in the calendar. 4. Clicking the close button will close the calendar display and take you back to the main page.
Special requirement	Supervisor is successfully logged into the system.

10 Plan Approval

Our client representative, the Health Care Agency personnel should approve this plan.