MASTER TEST PLAN

1. Test Plan Identifier JISS_Test_plan

2. References

- SRS submitted as a part of Assignment
- Project Description
- https://jmpovedar.files.wordpress.com/2014/03/ieee-829.pdf

3. Introduction

This is the Master Test Plan for the Judiciary Information System. This plan will mainly address the allocation of new cases, both directly and indirectly affected elements will be addressed. The primary focus will be the allocation of new cases by the registrar and the accessibility of those cases by judge and the lawyer, Also the classification of the cases based on different options given by the registrar. The project will have three levels of testing, Unit, System/Integration and Acceptance. The details for each level are addressed in the approach section and will be further defined in the level specific plans, and test cases for all JISS-related elements, including details for each unit test.

4. Test Items

4.1 Judge Class

- resolved_case_details_cin()
- resolved_case_deatils_keyword()

4.2 Lawyer Class

- resolved_case_details_cin()
- resolved_case_deatils_keyword()
- due_credits()

4.3 Registrar Class

- pendingcases()
- casescreate()
- resolvedcases()
- hearings()

- case_status()
- update_hearing()
- case_resolved()

5. Features to be Tested

6.1 Judge Class features

a) Sign in interface

b) Search by Cin

c) Log out interface

6.2 Lawyer Class features

a) SIgn in interface

b) Search by cin

c) Log out interface

6.3 Registrar Class features

a) SIgn in interface

b) Create a New case

c) Display pending cases

d) Resolved cases in a particular period

e) Allot Hearing date to a case having a unique Cin

f) Search the details of a particular case by Cin

g) Update hearing details of a particular case

h) Update Judgment Summary

i) Log out interface

6. Features not to be Tested

The following is a list of the areas that will not be specifically addressed. All testing in these areas will be indirect as a result of other testing efforts.

6.1 Create/Delete new users

The registrar is supposed to create the login details of new Judge and Lawyer and delete the existing ones using the default admin dashboard which django provides rather than creating them once he is logged in

6.2 Payment status of Lawyer

Whenever the Lawyer searches for a new case, The amount of money which he should pay for accessing the data must increase but this feature is not visible currently

6.3 Search by Keyword

The search of existing cases is done with the help of CIn , We have not yet implemented the search by keyword for finding cases, in case the keyword is a part of the case details

7. Approach :

7.1 Testing levels :

- Unit testing and system/integration testing will be done by the developers.
- Acceptance testing will be done by developers along with TAs and Professors guiding us through this project.

7.2 Test tools :

- For low-level testing (of the source code), a python compiler and an internet connection is required.
- For testing the website once it is hosted, the only requirement is a strong internet connection, and can be done by developers as well as non-developers.

7.3 Meetings :

- The developer team conducts meetings every day in order to keep each other up-to-date and speedily try to achieve all the goals and deadlines provided in this project.

The detailed methods for how exactly the test plans are designed, along with appropriate exception flow are explained below.

8. Testing plan and process

We will be providing outputs for all unit tests, which need to be matched for a unit test to be considered as “Passed”, any other result would be treated as “Failed”.

The following testing process will be used to test the Software :

- **Organize Project:** Creating a system test plan and approach
- **Design System Test:** Identifying the general error, Entrance and Exit Criteria, Expected Results etc. SRC document is used for the test conditions.
- **Designing Test Procedures:** Error Management and reporting of the status of program
- **Building Test environment:** Building the Hardware, software and data setups
- **Execute System Test:** The tests which were intended will be executed and the resulting discrepancies will be given to the Back End Development team.

Test Plans and Scenarios

Based on what is tested the test procedure is done in 3 parts

• Unit Testing

• Integration Testing

• System Testing

The template for designing these are as follows :

Unit Testing

Class MyUsers:

1. Username: String(must be non-empty)
2. Password: String(must be non-empty)
3. is_registar: Boolean(default false)
4. Is_judge: Boolean(default false)
5. is _lawyer: Boolean(default false)

Whenever a constructor is called, the program verifies that all the initialization parameters satisfy the above conditions. If one or more conditions are not satisfied, the constructor must raise an error and print the corresponding details of the error.

Test plan:

1. Call the constructor with a valid initialization parameter and print the object’s attributes.
2. Call the constructor with an invalid initialization parameter.

Class user_details:

1. Username: String(must be non-empty)
2. Password: String(must be non-empty)

Whenever a constructor is called, the program verifies that all the initialization parameters satisfy the above conditions. If one or more conditions are not satisfied, the constructor must raise an error and print the corresponding details of the error.

Test plan:

1. Call the constructor with a valid initialization parameter and print the object's attributes.
2. Call the constructor with an invalid initialization parameter.

Class courtcases:

1. Cin : Integer (system-generated, unique)
2. defendant : String (must be non-empty)
3. defandant_adress : String (must be non-empty)
4. crimetype : String (must be nonempty)
5. date : Datefield (must be nonempty)
6. location : String (must be nonempty)
7. arresting_officer : String (must be nonempty)
8. when_arrested : String (must be nonempty)
9. status : Boolean (default True)
10. presiding_judge : String (must be non empty)
11. public_prosecutor : String (must be non empty)
12. lawyer : String (must be non empty)
13. starting_date : Datefield (must be nonempty)
14. expected_completion_date : Datefield (must be nonempty)
15. hearing_date : Datefield (must be nonempty)
16. judgement_date : Datefield (must be nonempty)
17. judgement_summary : String (must be non empty)

Whenever a constructor is called, the program verifies that all the initialization parameters satisfy the above conditions. If one or more conditions are not satisfied, the constructor must raise an error and print the corresponding details of the error.

Test plan:

1. Call the constructor with a valid initialization parameter and print the object's attributes.
2. Call the constructor with an invalid initialization parameter.
3. Once the registar allots a hearing date, It must be displayed over here
4. If the registrar updates the hearing details, It must be reflected in the courtcases class also

5. If the registrar decides to update the judgement summary after a new hearing , it must be reflected in this class

Class hearing_details:

1. Cin : Integer(must be non-empty)
2. Date : DateField(must be non-empty, in the format: yy-mm-dd)
3. hearing_summary : String(must be non-empty)

Whenever a constructor is called, the program verifies that all the initialization parameters satisfy the above conditions. If one or more conditions are not satisfied, the constructor must raise an error and print the corresponding details of the error.

Test plan:

1. Call the constructor with a valid initialization parameter and print the object's attributes.
2. Call the constructor with an invalid initialization parameter.
3. When the registrar updates the hearing details, this class must also get updated

Class period:

1. starting_date : Datefield(must be non-empty, in the format: yy-mm-dd)
2. end_date : Datefield(must be non-empty, in the format: yy-mm-dd)

Whenever a constructor is called, the program verifies that all the initialization parameters satisfy the above conditions. If one or more conditions are not satisfied, the constructor must raise an error and print the corresponding details of the error.

Test plan:

3. Call the constructor with a valid initialization parameter and print the object's attributes.
4. Call the constructor with an invalid initialization parameter.

Class judgement:

1. Cin : Integer (must be non-empty)
2. judgement_date : Datefield (must be non-empty, in the format: yy-mm-dd)
3. attending_judge : String (must be non-empty)
4. judgement_summary : String (must be non-empty)

Whenever a constructor is called, the program verifies that all the initialization parameters satisfy the above conditions. If one or more conditions are not satisfied, the constructor must raise an error and print the corresponding details of the error.

Test plan:

1. Call the constructor with a valid initialization parameter and print the object's attributes.
2. Call the constructor with an invalid initialization parameter.
3. The judement class must be updated if the registrar decides to update any of its attribute details

Class Judge:

1. Inherited from class MyUsers, so super parameters checked by Class MyUsers

The constructor must always check that the initialization parameters satisfy the above conditions. It must raise an error and print the corresponding message in case of invalid initialization parameters.

Test Plan

1. Input the correct login details of Judge and Log In to the dashboard of the Judge Panel and displaying "Hello Judge"
2. The Judge should have an option to search the past cases by Cin, First enter a invalid Cin, No past cases should be displayed, next the Cin of a case which has not yet been completed must be entered, this must also not result in the view of the case, Lastly the Cin of a case which has a judgement day must be entered, this must display the case details of this particular case

Class Lawyer:

1. Inherited from class MyUsers, so super parameters checked by Class MyUsers

The constructor must always check that the initialization parameters satisfy the above conditions. It must raise an error and print the corresponding message in case of invalid initialization parameters.

Test Plan

1. Input the correct login details of Lawyer and Log In to the dashboard of the Lawyer Panel and displaying "Hello lawyer"
2. The Judge should have an option to search the past cases by Cin, First enter a invalid Cin, No past cases should be displayed, next the Cin of a case which has not yet been completed must be entered, this must also not result in the view of the case, Lastly the Cin of a case which has a judgement day must be entered, this must display the case details of this particular case.

3. For every case which the lawyer searches, the amount of Due Credits must be increased, and initially the number of due credits must be 0 when no prior searches have been done, This has yet to be implemented completely

Class Registrar:

1. Inherited from class Myusers, so super parameters checked by Class Myusers

The constructor must always check that the initialization parameters satisfy the above conditions. It must raise an error and print the corresponding message in case of invalid initialization parameters.

Test Plan

1. Input the correct login details of registrar and Log In to the dashboard of the registrar Panel displaying “hello registrar”
2. Test the new_case function by inputting the details of a new case, First test out by giving dates in the invalid format, Then an error will pop up to enter the dates in the correct format
3. Test the pending cases function, It must display all the current pending cases, and if none of the cases are pending then it will just show a blank column with no cases displayed
4. Test the resolved_cases features by first inputitng dates in a invalid format which should result in the display of an error and once we enter the correct start and end dates, The cases in this period must be displayed
5. Check the Allot_hearing_date feature by first inputting a invalid date which should result in the display to show to enter a valid date and if the date is alloted to a invalid cin, no new cases are formed , and when done to a valid cin but result in the appearance of the hearing date when the case is searched by cin
6. The Judge should have an option to search the past cases by Cin, First enter a invalid Cin, No past cases should be displayed, next the Cin of a case which has not yet been completed must be entered, this must also not result in the view of the case, Lastly the Cin of a case which has a judgement day must be entered, this must display the case details of this particular case
7. Check the allot_judgement_summary feature by allotting a valid judgement date for a case with valid cin and then search for the case with the same cin, this should result in the display of the case with the updated details, when an invalid date is posted, a popup must be shown which would display to enter a valid date

Incremental Testing

GUI Testing

- This is the test to see that all the interface buttons are working properly i.e., login button actually logs the user in etc.
- Checked basic **GUI Elements** like Buttons, Radio Buttons, CheckBoxes, TextBoxes, DropDown Menus
- Checked common **GUI Features** like logout button, Home button etc

9. Item pass/fail criteria :

- If the outputs match the provided golden outputs, then it is considered as “Passed”, otherwise “Failed”.

- Proper exception classes will be provided to throw and resolve different kinds of exceptions.
- Efficacy will be based on % of tests passed.

10. Suspension criteria and resumption requirements :

The testing of the software will be suspended based on the following criteria :

1. The actual outputs do not match the golden (expected) outputs.
2. A fatal and unexpected error occurs, such as crashing of the server or database.
3. The website stops responding to user interaction altogether.

The testing of the software is resumed ONLY after all the above issues have been successfully resolved and do not occur again.

No amount of bugs/ defects is acceptable in the delivery of the final software.

11. Test deliverables :

- Test plan
- Test suite

12. Remaining test tasks :

TASK	Assigned To	Status
Create Test Plan	Dev Team	Finished
Define Unit Test Cases	Dev Team	Finished
Verify prototypes of Screens	Dev Team	Pending
Verify prototypes of Reports	Dev Team	Finished

13. Environmental Needs :

The following elements are required for overall testing effort for the Online Sales Portal:

- For low level testing access to python 3 compiler is required, for making the codes and repeated testing of it.
- For high level testing access to an up to date web browser and robust internet connection is required.
- Test data (test cases) have been designed manually by the developers, such that all features and sections of the software are exhaustively tested, to find out and eliminate any bugs whatsoever.

14. Staffing and training needs :

14.1 Training given to users

- The Managers need some prior training on the portal to navigate it easily. However, no special sources are required. All explanations will be provided in the help section available in the portal itself (after logging in).
- The Sellers and Buyers do not need any prior training to use the website. Reading the help section and policies carefully will be sufficient to effectively use the portal.

14.2 Training given to developers

Any developer who wishes to do detail testing of the software, will require to have experience in the following fields :

- Python 3 language
- PyMongo library (for database)
- Flask library (for backend)
- CSS, Javascript, HTML, Bootstrap (for frontend)

15. Responsibilities :

	Dev Team	Client
System Integration test execution	X	
Unit test Documentation and execution	X	
System Design Reviews	X	X
Detail Design Reviews	X	
Test Procedure and rules		
Screen & Control Prototype Reviews	X	X

16. Schedule

Time has been allocated within the project plan for the following testing activities.

A. Review of Requirements document by the complete team (and then to be validated by the Professor and the TA) and initial creation of Inventory classes, sub-classes and objectives.
B. Development of Master test plan with time allocated for at least two reviews of the plan.
C. Integrating various modules involved and running System Integration tests.

D. Running various design related reviews to ensure that the design proposed will meet the specified requirements.
F. Unit test time within the development process.

G. Time allocated to correct any small discrepancies, with the help of feedback provided by the Professors and TAs.

17. Planning risks and contingencies

We have provided some contingency plans with regards to a few issues which might occur. They are as follows:

A. The payment done by the buyer does not directly go to the seller. It first stays with the management of OSP. When the buyer approves that he/she has received the product, money is transferred to the seller. This helps avoid any counterfeit seller and makes the portal more secure and reliable to use.

B. Random people cannot sign up as managers as they need to have a unique passcode to sign-up. This stops people from making fake manager accounts and disrupting the online portal.

18. Approvals

Development Manager 1	Bannuru Rohit Kumar Reddy
Development Manager 2	Chethan Krishna Venkat
Development Manager 3	Valugula Sai Varshith

TEAM MEMBERS :

- Valugula Sai Varshith - 21CS10073
- Chethan Krishna Venkat - 21CS30036
- Bannuru Rohit Kumar Reddy - 21CS30011