**Software Engineering Test Plan  
for *Coffee-oT***

**Written by Natch Sadindum – NS174 – 1269188  
SOFTFLUX**

# CONTENTS

**Contents Page ……………………………………………………………………..Page 2**

**1.1 – Testing Discussion ……………………………………………………….Page 3**

**1.2 – Tests ……………………………………………………………………………Page 3**

**REQ1 ………………………………………………………………………………………………………………………Page 3**

**REQ2 ………………………………………………………………………………………………………………………Page 3**

**REQ3 ………………………………………………………………………………………………………………………Page 4**

**REQ4 ………………………………………………………………………………………………………………………Page 4**

**REQ5 ………………………………………………………………………………………………………………………Page 5**

**REQ6 ………………………………………………………………………………………………………………………Page 5**

**REQ7 ………………………………………………………………………………………………………………………Page 6**

**REQ8 ………………………………………………………………………………………………………………………Page 6**

**REQ9 ………………………………………………………………………………………………………………………Page 6**

**REQ10 …………………………………………………………………………………………………………………….Page 6**

**REQ11 …………………………………………………………………………………………………………………….Page 7**

**IT1 ………………………………………………………………………………………………………………………….Page 8**

**IT2 ………………………………………………………………………………………………………………………….Page 8**

**IT3 ………………………………………………………………………………………………………………………….Page 8**

**2.1 – Testing Map – Requirements 1 - 6 ………………………………Page 9**

**2.2 – Testing Map – Requirements 1 - 6 ………………………………Page 10**

# 1.1 - Testing Discussion

The below tables of test cases are unit tests – aimed to test each individual component’s code. The unit tests listed cover all requirements of the program by checking that given set inputs, an expected. Most common code paths should be covered and some rare ones (such as T9 and related) will be covered.

Note that some of the tests are pseudo-**integration tests**, as some of the components are inter-reliant by their nature, where a failure in one component renders another useless.

Most of the tests can be done manually using a debugger to obtain data passing through the system, such as T14.

A few tests such as T3 and T17 should be tested autonomously using a script. These tests target different input types or require large numbers of objects to be created and compared.

Directly-typed user input is limited to the login/signup stage by design. This limits the opportunity for boundary value testing, as most input methods themselves are limited due to the frontend design. Equivalence partitioning is doable, as typed input can be grouped into expected, invalid and extreme values (such as high Unicode values not normally used, common letters and numbers and low ASCII characters not normally used)

Most of the tests are done via white-box testing, as we need to test the limits of our software with knowledge of how it could be broken in everyday use. Black-box testing can be done after unit testing is complete, in a similar track to the white-box tests, as a simulation of everyday use.

# 1.2 - Tests

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test ID | Scenario | Purpose | Expected Result | Notes |
| T1 | Access other functions aside from login/signup as unauthorised user | Test access to pages apart from login | User is redirected to unauthorised access page, then to login page | Manually testable, only variable is logged-in user type |
| T2 | Access any functions aside from login/signup page as authorised user | User is moved to post-login page for access to other functions |
|  |  |  |  |  |
| T3 | Create a new user | Test if user profiles can be created using a variety of Unicode input types | New user inserted into database | Should be automated, to test different input types in each of the fields.  Eg. Normal expected input (standard names), high Unicode (훊) and low ascii (alt+00) |
| T4 | Create a new user using invalid (non-null) data | Error message shown, no profile created | Code should check and restrict user input types as defined by the program requirements/specification |
| T5 | Create a new user profile with empty fields |
|  |  |  |  |  |
| T6 | Log in using valid user details of an existing authorised user | Test login process for existing user | User details are checked against database  Successful login, user can now access other pages, user logged-in status updated | See T3 |
| T7 | Log in using invalid details | Test login process for boundary/invalid input | User details are checked against database  Unsuccessful login and error message | - |
| T8 | Log in with empty fields | - |
| T9 | Log in as a coffee machine user | It should not be possible to log in as a coffee machine using the login page |
| T10 | Log in as a long-previously created user | Test login process for repeated logins | Successful login |  |
| T11 | Log in as the same user on multiple devices |  |  |  |
|  |  |  |  |  |
| T12 | Log out of program while logged in as an authorised user | Test log out function. | Successful logout, user profile logged-in status updated.  All data relating to user should be deleted client-side upon logout (privacy) | Manually testable, should be done on different device types. |
| T13 | Log out of program while logged in as a coffee machine |  | Any user should not be able to log out a coffee machine this way, must be done by the coffee machine register function |
|  |  |  |  |  |
| T14 | Add new Coffee Machine record while logged in as an authorised user | Check creation and association of Coffee Machine Record logs | New coffee machine record created, added to user account in the database |  |
| T15 | Add new Coffee Machine record while logged in as a coffee machine |  |  | Coffee machine user type should not be able to create records |
| T16 | Add new Coffee machine record while logged in as authorised user | Test pseudorandom key generator.  Test overlap fixing code | New key generated  No overlap of pseudorandom key with any existing keys | As unique key is 16-characters long, it is extremely unlikely that there will be any overlaps when generating a key. Assuming any case-insensitive letter and number permutation, there are 1.529\*1023 permutations. |
| T17 | Add multiple identical new Coffee machine records while logged in as authorised user | Test handling of identical logs | Only one identical log created | Code should be designed to prevent duplicates from being generated |
|  |  |  |  |  |
| T18 | Delete a Coffee Machine record while logged in as an authorised user | Test deletion of records | If records exist on a user, then the targeted record is deleted.  If no records exist, then no operation occurs | REQ6 code should handle deletion of non-existent events |
| T19 | Delete a Coffee Machine record while logged in as a coffee machine |  |  |
| T33 | Delete a coffee machine record on multiple devices using the same user | Test simultaneous functionality | First action should delete a record and update immediately, preventing other devices from deleting the missing record |  |
|  |  |  |  |  |
| T20 | View Calendar while logged in as an authorised user | Test if activation events associated with an authorised user are loaded/stored | All activation events associated with the user should be loaded | Perform after adding and deleting activation events successfully |
| T21 | View Calendar while logged in as a Coffee Machine | No events loaded |  |
|  |  |  |  |  |
| T22 | Zoom Calendar while logged in as authorised user with existing calendar events | Test if calendar events are loaded/unloaded successfully upon zooming  Test if calendar zooms properly between year/month/week/day scale | Calendar events are loaded/unloaded successfully from database and moved to frontend for display |  |
| T23 | Zoom Calendar while logged in as Coffee Machine |  |  |
|  |  |  |  |  |
| T24 | Add new Activation event as authorised user | Test if activation events are added with correct date/time and coffee machine association | New activation events are added to database and viewable via REQ7/REQ8 |  |
| T25 | Add new Activation event as coffee machine |  |  |
|  |  |  |  |  |
| T26 | Delete Activation event as authorised user | Test deletion of events from database | Targeted activation event is deleted from database | Due to frontend, only events that exist can be deleted |
| T27 | Delete Activation event as coffee machine |  |  |
| T28 | Delete Activation event when user has no events | Option to delete events should not be accessible if none exist |
|  |  |  |  |  |
| T29 | Test polling of registered coffee machine with scheduled activation event within 30 seconds of test with authorised user | Test polling function | “BREW!” response from coffee machine | Should be tested with one or more activation events and users associated with the coffee machine. Should be tested with multiple coffee machines. |
| T30 | Test polling of registered coffee machine without any activation events |  | No response |  |
| T31 | Test polling of registered coffee machine with multiple scheduled activation events from different users within 30 seconds of test with authorised user | Test polling function against many activation events | “BREW!” response from coffee machine | Need to see how many events can be detected and acted upon by coffee machine polling. |
| T32 | Test polling of registered coffee machine with invalid scheduled activation events | Test polling function vs invalid activation event | No response | Should only provoke response if all parameters match. Potential data corruption test. |

The following tests target multiple components at once and are performed in a black-box manner. Given that each of the components now work individually, these integration tests are intended to simulate everyday use.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test ID | Scenario | Purpose | Expected Result | Notes |
| IT1 | - Create a user  - Log in immediately with that user  - Add 3 coffee machine records to the user  - Delete one of the coffee machine records from the user  - Add 3 activation events, one for each registered coffee machine, to activate within 2 minutes  - Delete one activation event  - log out | Test a number functions in an “everyday” setting | Successful user creation and login, 2 coffee machine records in user at end of test, 2 BREW! responses from 2 coffee machines at end of test, successful logout. | Standard expected input is to be used for any section requiring typed user input.  Covers all requirements visible to a user during normal use |
| IT2 | - create a user  - log in immediately  - log out  - wait for one minute, attempt to log in as the same user created earlier  - carry on from instruction 3 of IT1 | Same as IT1 | Same as IT1, except testing logging in well after a user is created. |
| IT3 | - create a user  - log in with the same user on at least 3 devices  - carry on from instruction 3 of IT1 for each device | If unique coffee machines are added and deleted, then same as IT1 | Same as IT1.  Synchronisation test, not expected to occur realistically, as an authorised user should not be able to log in while logged in. |

# Testing Map – Requirements 1 - 6:

T19

T33

T18

REQ6

T16

T17

REQ5

T15

T14

T13

T12

REQ4

REQ3

T11

T10

T7

T8

T9

T6

T5

T4

T3

REQ2

T2

T1

REQ1

2.2 Testing Map – Requirements 7 - 11All the following requirements are closely linked and rely on each other to function correctly.

T3

T30

T31

T32

T28

REQ11

T27

T28

T26

REQ10

REQ9

T24

T25

T20

T21

REQ7

REQ8

T22

T23

REQ2