Test Plan for Quarters

James Anthony (anthonjb)
Wenqiang Chen (chenw25)
Carolyn Chong (chongce)
Kevin Ly (lyk2)

October 29, 2015

Contents

Revision History

Date	**Comments**
October 21, 2015	Created first draft.

Template

This document makes use of the Software Test Plan (STP) Template for all of its organization.

1 Acronyms and Definitions

Acronym	Description
PoC	Proof of Concept

2 Plans for Automated Testing

Though some tests may be done manually for the PoC Demo, all tests will be automated for the Final Demo in April. We will write our own test scripts.

3 Plans for Unit Testing

QUnit will be used for unit testing. QUnit is a Javascript unit testing framework.

4 System Tests

[Use this as a template. Create a new subsection for each feature to be tested. —CC]
Test Type: Structural/Functional/Unit, Static/Dynamic, Manual/Automated.
Test Factors: Correctness/Learnability/Maintainability/Reliability.
Tools Used: unit testing framework, code coverage metrics, static checkers, automated testing, load testing (like JMeter), etc.
Schedule: PoC Demo November 16 / Final Demo April 1.
Team Member Responsible:
Methodology: The "how".

Test Case	Initial State	Input	Output
4.			

4.1 User Registration

Test Type: Functional/Static/Dynamic/Manual/Automated.
Test Factors: Correctness, Reliability.
Tools Used: Google reCAPTCHA.
Schedule: Begin static testing November 6. Complete manual dynamic tests by PoC Demo November 16. Complete automated dynamic tests by Final Demo April 1. Testing for creating an account with Google or Facebook will be completed for the Final Demo April 1.
Team Member Responsible:
Methodology: The main objective of user registration is to create a user account to be used for login. Users must use a valid email address and pass a user identification procedure.

This ensures the user is human and prevents spam and automated scripts from accessing the application and abusing its services. Alternatively, a user can create an account using an existing Google or Facebook account. Testing is manual and automated. Manual testing involves people manually going through the registration process in real-time as a user. Automated testing involves systemically attempting SQL injections to test for valid and invalid registrations. Google reCAPTCHA validates that users are legitimate.

Test Case	**Initial State**	**Input**	**Output**
4.1.1	Registration page. Empty fields.	Valid values entered and passes reCAPTCHA test.	User values are correctly stored in database. Redirected to application home page.
4.1.2	Registration page. Empty fields.	Empty field(s).	Stays on the same page. Error message appears: "Missing field". Empty field is highlighted.
4.1.3	Registration page. Empty fields.	Invalid email address.	Stays on the same page. Error message appears: "Invalid email". Email field is highlighted.
4.1.4	Registration page. Empty fields.	Email address already stored in database.	Stays on the same page. Error message appears: "An account with this email has already been created". Email field is highlighted.
4.1.5	Registration page. Empty fields.	Fails reCAPTCHA test.	Stays on the same page. reCAPTCHA error message appears. Test field is highlighted.
4.1.6	Registration page. Empty fields.	Selects Register with Google Account.	User values are correctly stored in database. Redirected to application main page.
4.1.7	Registration page. Empty fields.	Selects Register with Facebook Account.	User values are correctly stored in database. Redirected to application main page.

4.2 User Login

Test Type: Functional/Static/Dynamic/Manual/Automated.
Test Factors: Correctness, Reliability.
Tools Used:
Schedule: Begin static testing November 6. Complete manual dynamic tests by PoC Demo November 16. Complete automated dynamic tests by Final Demo April 1. Testing for logging in with a Google or Facebook account will be completed for the Final Demo April 1.
Team Member Responsible:
Methodology: The main objective of user login is to ensure a secure process where only valid users are allowed to enter the application. Testing involves authenticating users against an existing database to determine if they are valid users or not. Testing is manual and automated. Manual testing involves people manually going through the login process in real-time as a user. Automated testing involves systemically attempting SQL injections to test for valid and invalid logins.

Test Case	Initial State	Input	Output
4.2.1	Login page. Empty username and password fields.	Valid username and password combination.	Redirected to application main page.
4.2.2	Login page. Empty username and password fields.	Invalid username.	Stays on the same page. Error message appears: “Invalid username”. Fields are highlighted.
4.2.3	Login page. Empty username and password fields.	Valid username and invalid password.	Stays on the same page. Error message appears: “Incorrect password”. Password field is highlighted.
4.2.4	Login page. Empty username and password fields.	Empty username and/or password fields.	Stays on the same page. Error message appears: “Missing field”. Fields are highlighted.
4.2.5	Login page. Empty username and password fields.	Selects Log in with Google Account	Redirected to application main page.
4.2.6	Login page. Empty username and password fields.	Selects Log in with Facebook Account.	Redirected to application main page.

4.3 Calendar

Test Type: Functional/Static/Dynamic/Manual/Automated.
Test Factors: Learnability, Reliability.
Tools Used: [todo —CC].
Schedule: Begin static testing November 6. Complete manual dynamic tests by PoC Demo November 16. Complete automated dynamic tests by Final Demo April 1. Tests for the syncing of a user's personal calendar will be completed for the Final Demo April 1.
Team Member Responsible:
Methodology: The Calendar feature allows users to add/delete events and chores to a shared Calendar between members of a house. This shared Calendar can be synched with a user's personal Calendar. Testing is manual and automated. Manual testing involves a person manually going through the process of adding/deleting an event or chore to the Calendar in real-time as a user, and then checking if those updates are properly synched with the user's personal Calendar. Automated testing involves unit testing each possible input.

Test Case	Initial State	Input	Output
4.3.1	Calendar page. Empty form.	Add event/chore. Correct values entered in fields.	Form closes. Event/chore is added to database. Event/chore is updated on Calendar.
4.3.2	Calendar page. Empty form. [check —CC]	Add event/chore. Incorrect values entered in fields.	Form remains open. Error message appears. Incorrect fields are highlighted.
4.3.3	Calendar page. Empty form.	Add event/chore. Empty field(s).	Form remains open. Error message appears: "Missing field". Empty fields are highlighted.
4.3.4	Calendar page.	Click button to delete event/chore.	Event/chore is removed from database. Event/chore is no longer displayed on Calendar.

4.4 Maintenance Tracking

Test Type: Functional/Static/Dynamic/Manual/Automated.
Test Factors: Correctness, Reliablility.
Tools Used: QUnit, Chron Scripts.
Schedule: Begin static testing November 6. Complete manual dynamic tests by PoC Demo November 16. Complete automated dynamic tests by Final Demo April 1.
Team Member Responsible:

Methodology: The maintenance tracking system allows tenants to create maintenance requests, where the landlord then responds and updates with further information. This portions of this system is restricted based on the user type; tenants cannot modify maintenance ticket properties. This component will be tested using unit test for functionality, with automated testing to ensure the permissions are handled properly. Static database checkers will be used in conjunction with the automated test cases to check for proper database modifications.

Test Case	**Initial State**	**Input**	**Output**
4.4.1	Quarters Web Application	Open maintenance system	Maintenance system opens and shows new maintenance tickets with existing tickets in chronological
4.3.2	Maintenance System.	Clicks on maintenance ticket	Inner dialog opens displaying all properties in a maintenance ticket.
4.3.3	Maintenance System	Entering a search query or adding a filter	sort and filter maintenance tickets and reveal only successful tickets.
4.3.4	Maintenance Ticket Window	Modifying properties of a ticket	save icon appears in dialog to confirm changes
4.3.5	Maintenance Ticket Window	Saving ticket properties	window will close, and database will be updated to reflect changes
4.3.6	Maintenance Ticket Window	Deleting Ticket	confirmation window will appear, upon deletion confirmation: close window and remove data from database
4.3.7	Maintenance System	click on create new request	Opens a new ticket window
4.3.8	New Maintenance ticket window	click on create empty fields	window will remain opening, prompt will display error message
4.3.9	New Maintenance ticket window	Click on create, required fields filled	window closes, database will be updated with new ticket
4.3.10	New Maintenance ticket window	click on cancel with fields filled	window remains open, prompt will ask for confirmation on close

4.3.11	Confirmation Prompt	click on OK	closes prompt and dialog
4.3.10	Confirmation Prompt	click on cancel	closes prompt, dialog remains open

4.5 House Management

Test Type: Functional/Static/Dynamic/Manual/Automated.
Test Factors: Learnability, Reliability.
Tools Used: [todo —CC].
Schedule: Begin static testing November 6. Complete manual dynamic tests by PoC Demo November 16. Complete automated dynamic tests by Final Demo April 1.
Team Member Responsible:
Methodology: House management system is the feature which allows user to view and modify information in regards to the house and create and delete houses. Unit tests can be created for each function in the feature which will be included in the automated testing sequence.

Test Case	Initial State	Input	Output
4.5.1	House Management, not admin user	Click modify information	Nothing
4.5.2	House Management, admin user	Click modify information	input fields become editable
4.5.3	House Management, admin	modify information fields	save button opens, discard changes appears.
4.5.4	House Management, any user	click on View Documents	redirects to new page showing all uploaded documents in House
4.5.5	House Documents, any user	clicks on a document	retrieves documents and initiates file transfer
4.5.6	House Documents, admin	clicks on Add Documents	upload window opens for user upload, file will be transfer to server and information is updated in database
4.5.7	House Documents, admin	clicks on delete document	prompt opens

4.5.8	deletion prompt, admin	clicks on yes	prompt closed, file is removed from display, database is updated
4.5.9	deletion prompt, admin	clicks on no	prompt closed
4.5.10	House Management, any user	clicks on view members	shows all memebers of the house and their role
4.5.11	House Management, admin, members list visible	clicks on add member	Dialog will appear
4.5.12	Member Dialog, admin, fields empty	clicks on ok	prompt opens, notifying missing fields
4.5.13	Member Dialog, admin, fields complete	clicks on ok	window closes, new user is notified, database is updated, member status pending
4.5.14	Member Dialog, admin,	clicks on cancel	window closes

4.6 Live Chat

Test Type: Functional/Static/Dynamic/Manual/Automated.
Test Factors: Correctness, Reliability.
Tools Used:
Schedule: Begin static testing November 6. Complete manual dynamic tests by PoC Demo November 16. Complete automated dynamic tests by Final Demo April 1.
Team Member Responsible:
Methodology: The main objective of live chat is to allow another means of communication inside the house; it will replace the use of Facebook messenger and text message so the user does not have to switch between applications. The testing involves one user establishing live chat with another user. Testing will be manual and automated. Manual testing involves one user(A) sending a message to other user(B) and ensuring user(B) receives the message without delay. Automated testing involves a macro which simulates the sending process.

Test Case	Initial State	Input	Output
4.9.1	Main page. User logged in.	Click on user to send message	Chat window opens, with target user's name as window name.
4.9.2	Main page. User logged in.	With chat window open, insert text and click send	Chat windows shows user's message and the time stamp it's sent.
4.9.3	Main page. User logged in.	With chat window open, do not insert text and click send	Chat windows does not change, nothing is sent.
4.9.4	Main page. User logged in.	Click on user with notification pending.	Chat window opens with target user's message displayed with it's sent time stamp.

4.7 Notification

Test Type: Functional/Static/Dynamic/Manual/Automated.
Test Factors: Correctness, Reliability.
Tools Used:
Schedule: Begin static testing November 6. Complete manual dynamic tests by PoC Demo November 16. Complete automated dynamic tests by Final Demo April 1.
Team Member Responsible:
Methodology: The main objective of notification is to remind user of events that has had happened; users should be notified immediate after the event has taken place. The testing involves one user completing different actions which generates notification and have another user related to this event receive notification. Automated testing involves a macro which simulates the sending process.

Test Case	Initial State	Input	Output
4.10.1	Main page. User(A) logged in.	User(B) sends money request	User(A) sees notification of pending payment due.
4.10.2	Main page. User(A) logged in.	User(A) pays user(B)	User(B) sees notification of payment completed.
4.10.3	Main page. User(A) logged in.	User(A) has late payment	User(A) sees notification of late payment.

4.10.4	Main page. User(A) logged in.	User(A) joins a house	Other users in that house sees notification that user(A) joined the house.
4.10.5	Main page. User(A)(landlord) logged in.	User(B) sends maintenance ticket(Critical)	User(A) sees notification of unresolved maintenance ticket, receives email, receives text message
4.10.6	Main page. User(A)(landlord) logged in.	User(B) sends maintenance ticket(Major)	User(A) sees notification of unresolved maintenance ticket, receives email
4.10.7	Main page. User(A)(landlord) logged in.	User(B) sends maintenance ticket(Minor)	User(A) sees notification of unresolved maintenance ticket
4.10.8	Main page. User(A) logged in.	User(B)(Landlord) resolves a maintenance ticket	User(A) sees notification of resolved maintenance ticket
4.10.9	Main page. User(A) logged in.	User(B) sends user(A) a message	User(A) sees notification of unread message
4.10.10	Main page. User(A) logged in.	User(B) makes a post in bulletin board	User(A) sees notification of unread post
4.10.11	Main page. User(A) logged in.	User(B) replies to a post made by user(A)	User(A) sees notification of unread reply

4.8 Administrative File Storage

Test Type: Mantual/Automated.
Test Factors: Correctness, Security, Reliability.
Tools Used:
Schedule: After proof of concept. **Team Member Responsible:**
Methodology: A script can be used to test the process of uploading and downloading multiple files of different types and sizes.

Test Case	**Initial State**	**Input**	**Output**
0.0.0	0 files in storage.	User tries to upload a file of size s, where $s \leq$ max file size.	Successful file upload.

0.0.1	0 files in storage.	User tries to upload a file of size s, where $s >$ max file size.	Error message indicating file has not been uploaded.
0.1.0	n files in storage.	User tries to upload a file of size s, where $s \leq$ total remaining space.	Successful file upload.
0.1.1	n files in storage.	User tries to upload a file of size s, where $s >$ total remaining space.	Error message indicating file has not been uploaded.
0.1.2	n files in storage.	User tries to upload a file with an invalid type.	Error message indicating file has not been uploaded.
0.1.3	n files in storage.	User requests to download a file.	Successful file download.
0.1.4	n files in storage.	Connection interrupted while download is in progress.	Error message indicating file has not been downloaded.

4.9 Bulletin Board

Test Type: Mantual/Automated.
Test Factors: Correctness, Reliability.
Tools Used:
Schedule: After proof of concept. **Team Member Responsible:**
Methodology: A script can be used to test the process of posting on the discussion board, and commenting on existing posts.

Test Case	**Initial State**	**Input**	**Output**
1.0.0	No posts on bulletin board.	A post with 0 characters	Empty post is disgarded and not added to bulletin board.
1.0.1	No posts on bulletin board.	A post with n characters, where $n > 0$.	Bulletin board is updated with the post of n characters.
1.1.0	p posts on bulletin board, where $p > 0$.	A post with 0 characters	Empty post is disgarded and not added to bulletin board.
1.1.1	p posts on bulletin board, where $p > 0$.	A post with n characters, where $n > 0$.	Bulletin board is updated with the post of n characters.

1.1.2	p posts on bulletin board, where $p > 0$.	A comment with 0 characters on an existing post p.	Empty comment is disgarded and not added to bulletin board.
1.1.3	p posts on bulletin board, where $p > 0$.	A comment with n characters where $n > 0$, on an existing post p_i.	Comment is added to the list of comments associated with post p_i.

4.10 Finance

Test Type: Manual
Test Factors: Security, Correctness, Reliability.
Tools Used:
Schedule: After proof of concept. **Team Member Responsible:**
Methodology: Tests cann be performed by having one user add payment deadlines, and having another user send arbitrarty amounts of money via PayPal. Speed and accuracy of transactions can be tracked. Transaction records can be manually evaluated for correctness.

Test Case	**Initial State**	**Input**	**Output**
2.0.0	No payments due.	User initiates PayPal transaction.	Error message indicating that no payments are due the current time.
2.0.1	No payments due.	User posts payment request with some deadline.	New payment deadline added.
2.1.0	Payment due.	User initiates PayPal transaction.	Transaction is handled by PayPal. All users involved are notified of the completed payment. Deadline is removed from list of current payments due.
2.1.1	Payment due.	Payment has not been completed, and deadline has passed.	All users involved are notified that the deadline has passed. Deadline is marked as past due, and users will continue to be notified until either the payment has been completed, or the due payment is removed.
2.1.2	Payment due.	User who posted the original due payment removes the request.	Payment request is removed from the list of due payments.