Test Report for Quarters

Team 6
James Anthony (anthonjb)
Wenqiang Chen (chenw25)
Carolyn Chong (chongce)
Kevin Ly (lyk2)

March 20, 2016

Contents

1 Introduction **3**

2 Automated Testing **3**

3 System Tests **3**

4 System Tests **3**
4.1 User Registration 4
4.2 User Login 4
4.3 Calendar 5
4.4 Maintenance Tracking 6
4.5 House Management 6
4.6 Landing Page 8
4.7 Finance 8
4.8 Notifications 9
4.9 Administrative File Storage 11
4.10 Bulletin Board 11

5 Non-Functional Tests **12**
5.1 Look and Feel 12
5.2 Usability 13
5.3 Performance 13
5.4 Robustness 13

6 Summary of Changes **13**

List of Figures

List of Tables

Revision History

Date	Comments
March 20, 2016	Created first draft.

1 Introduction

This testing report shows the results of both system tests and non-functional tests on the Quarters application. The system tests are reported based on each individual module. Non-functional tests included tests on usability, performance and robustness.

2 Automated Testing

[Explain use of automated testing, or explain why it was not feasible for this project. —CC] Automatic testing is being used in this project to unit test various parts of the system. The project's components are broken up in to several parts: Client side javascript components, Server side access and security.

The client side javascript is tested every week once a week on an alternate web server using QUnit, a javascript unit-testing framework. Unit tests were written for each component to ensure every method does their intended action. The unit tests are rigorously tested to ensure all exceptions are handled.

Server Side access is tested via a python web crawler. To ensure that ever page is reachable. This is ran once a week similarly to the client side tests. The web crawler also crawls through all available link in a demonstration and checks for broken links. Security also tested within the web crawler by crawling with authentication and without authentication. A predetermined set of pages can only be accessible with out authentication such as the landing page, login and registration.

The unit tests will be updated and modified as more development continues. In practice these unit tests should ensure that updates to the code base and other changes to the server end do not break the system.

3 System Tests

[Specific system tests. All tests should be fully summarized in terms of initial state, input and expected output. Tests should be named. In cases where there are many similar tests just summarize the results. Provide enough info that someone could reproduce your tests. Provide traceability to test plan by referring to test case numbers or modules. —CC]

4 System Tests

[Specific system tests. All tests should be fully summarized in terms of initial state, input and expected output. Tests should be named. In cases where there are many similar tests just summarize the results. Provide enough info that someone could reproduce your tests.

Provide traceability to test plan by referring to test case numbers or modules. —CC] In this section the test cases carried out on each individual module are described. Trivial cases for some modules are not explicitly written out but instead described at a high level. Additional details are provided when necessary.

4.1 User Registration

No.	Test Case	Initial State	Input	Expected Output	Actual Output	Result
1.1	User Registration	Landing page. Empty fields.	Email and password entered. Clicks register.	Redirected to application main page.	As expected.	PASS
1.2	User Registration	Landing page. Empty fields.	Empty field(s). Clicks register.	Stays on the same page. Error message appears. Empty field is highlighted.	As expected.	PASS
1.3	User Registration	Landing page. Empty fields.	Email address already stored in database. Clicks register.	Stays on the same page. Error message appears. Email field is highlighted.	As expected.	PASS

4.2 User Login

No.	Test Case	Initial State	Input	Expected Output	Actual Output	Result
2.1	User Login	Landing page. Empty username and password fields.	Valid username and password combination. Clicks login.	Redirected to application main page.	As expected.	PASS
2.2	User Login	Landing page. Empty username and password fields.	Invalid username and password combination. Clicks login.	Stays on the same page. Error message appears. Fields are highlighted. After 5 unsuccessful attempts, user cannot login for 10 minutes.	As expected.	PASS

2.3	User Login	Landing page. Empty username and password fields.	Empty username and/or password fields. Clicks login.	Stays on the same page. Error message appears. Fields are highlighted.	As expected.	PASS
2.4	User Logout	Application main page.	Clicks logout.	User is successfully logged out from system. Redirected to login page.	As expected.	PASS
2.5	User Login	Landing page. Empty username and password fields. User attempting to login on another device while already logged in on a device.	Valid username and password combination. Clicks login.	Stays on the same page. Error message appears.	As expected.	PASS

4.3 Calendar

No.	**Test Case**	**Initial State**	**Input**	**Expected Output**	**Actual Output**	**Result**
3.1-3.4	Add event to Calendar.	Calendar page.	User selects date to add new event, enters information, clicks save.	Modal opens with fields, and closes upon save. Event is updated correctly on Calendar. The same output results if user selects existing event to modify.	As expected.	PASS

3.5	Delete event from Calendar	Calendar page.	User selects event to delete, clicks delete.	Modal opens with fields. Upon clicking delete, the modal closes and the event is removed from the Calendar.	As expected.	PASS

4.4 Maintenance Tracking

No.	Test Case	Initial State	Input	Expected Output	Actual Output	Result
4.1-4.5	Navigating to maintenance page	Quarters application.	User clicks on maintenance tab in the navigation bar	Application navigates to maintenance page, all maintenance tickets relevant to the house are shown	As expected	PASS
4.6	Delete ticket from maintenance page	Maintenance System.	User clicks on "X" button beside the maintenance ticket, user clicks confirm when confirmation window pops up.	confirmation window will appear. upon deletion conformation, close confirmation window, and ticket is removed from the page.	As expected	PASS
4.7-4.12	Create new maintenance ticket	Maintenance System.	User clicks on "new request".	Modal opens with fields, and closes upon save. New ticket is added into the page	As expected	PASS

4.5 House Management

No.	Test Case	Initial State	Input	Expected Output	Actual Output	Result

5.1	Modify house information, not admin.	House Management, not admin.	Click modify information.	Nothing.	As expected.	PASS
5.2	Modify house information as admin.	House Management, admin.	Click modify information.	Input fields become editable.	As expected.	PASS
5.3	Modify house information as admin.	House Management, admin.	Modify information fields.	Save button opens, discard changes appears.	As expected.	PASS
5.4	Modify house information as user.	House Management, any user.	Click on View Documents.	Redirects to new page showing all uploaded documents in House.	As expected.	PASS
5.5	Modify house information as user.	House Documents, any user.	Clicks on a document.	Retrieves documents and initiates file transfer.	As expected.	PASS
5.6	Modify house information as admin.	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.	As expected.	PASS
5.7	Modify house information as admin.	House Documents, admin.	Clicks on delete document.	Prompt opens.	As expected.	PASS
5.8	Modify house information as admin.	Deletion prompt, admin.	Clicks on yes.	Prompt closed, file is removed from display, database is updated.	As expected.	PASS
5.9	Modify house information as admin.	Deletion prompt, admin.	Clicks on no.	Prompt closed.	As expected.	PASS
5.10	Modify house information as user	House Management, any user.	Clicks on view members.	Shows all memebers of the house and their role.	As expected.	PASS

5.11	Modify house information as admin.	House Management, admin, members list visible.	Clicks on add member.	Dialog will appear.	As expected.	PASS
5.12	Modify house information as admin.	Member Dialog, admin, fields empty.	Clicks on ok.	Prompt opens, notifying missing fields.	As expected.	PASS
5.13	Modify house information as admin.	Member Dialog, admin, fields complete.	Clicks on ok.	Window closes, new user is notified, database is updated, member status pending.	As expected.	PASS
5.14	Modify house information as admin.	Member Dialog, admin.	Clicks on cancel.	Window closes.	As expected.	PASS

4.6 Landing Page

No.	**Test Case**	**Initial State**	**Input**	**Expected Output**	**Actual Output**	**Result**
6.1,6.2	Access login or registration.	Not logged in.	Clicks on login.	Modal opens and email and password fields appear. The same output results if user clicks on register.	As expected.	PASS

4.7 Finance

No.	**Test Case**	**Initial State**	**Input**	**Expected Output**	**Actual Output**	**Result**

7.6	Add a new bill to the house	Finance page	User clicks on "+" button, fills in all informations in modal window and click "save"	Modal window opens with fields, upon save with all fields filled in, a list of tenants that owes the user money will be added to the page	As expected.	PASS
7.7	Mark bill as paid	Finance page	User clicks on "Paid" button beside the bill, clicks ok on the confirmation window	Confirmation window will appear, upon clicking ok, the bill will have a ✓beside it	As expected.	PASS

4.8 Notifications

Test Type: Functional, Dynamic, Manual.
Tools Used: None.
Schedule: Begin testing after the PoC Demo. Complete automated tests by Final Demo April 1.
Team Member Responsible: Wenqiang Chen.
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.

Test Case	Initial State	Input	Output
8.1	Main page. User(A) logged in.	User(B) sends money request.	User(A) sees notification of pending payment due.
8.2	Main page. User(A) logged in.	User(A) pays user(B).	User(B) sees notification of payment completed.
8.3	Main page. User(A) logged in.	User(A) has late payment.	User(A) sees notification of late payment.
8.4	Main page. User(A) logged in.	User(A) joins a house.	Other users in that house sees notification that user(A) joined the house.

8.5	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.
8.6	Main page. User(A)(landlord) logged in.	User(B) sends maintenance ticket(Major.)	User(A) sees notification of unresolved maintenance ticket, receives email.
8.7	Main page. User(A)(landlord) logged in.	User(B) sends maintenance ticket(Minor).	User(A) sees notification of unresolved maintenance ticket.
8.8	Main page. User(A) logged in.	User(B)(Landlord) resolves a maintenance ticket.	User(A) sees notification of resolved maintenance ticket.
8.9	Main page. User(A) logged in.	User(B) sends user(A) a message.	User(A) sees notification of unread message.
8.10	Main page. User(A) logged in.	User(B) makes a post in discussion board.	User(A) sees notification of unread post.
8.11	Main page. User(A) logged in.	User(B) replies to a post made by user(A).	User(A) sees notification of unread reply.
8.12	Main page. User(A) logged in.	User(A) leaves a house.	Other users in that house sees notification that user(A) left the house.
8.13	Main page. User(A) logged in.	User(B) adds event to Calendar.	User(A) sees notification of added post.
8.14	Main page. User(A) logged in.	User(B) deletes event from Calendar.	User(A) sees notification of deleted event.
8.15	Main page. User(A) logged in.	User(A)has event happening on day.	User(A) sees notification of event.
8.16	Main page. User(A) logged in. Notification displayed.	User clicks on Notification icon.	Notification disappears.

4.9 Administrative File Storage

Test Type: Functional, Dynamic, Automated.
Tools Used: Custom Scripts.
Schedule: Begin testing after the PoC Demo. Complete automated tests by Final Demo April 1.
Team Member Responsible: James Anthony.
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
9.1	0 files in storage.	User tries to upload a file of size s, where $s \leq$ max file size.	Successful file upload.
9.2	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.
9.3	n files in storage.	User tries to upload a file of size s, where $s \leq$ total remaining space.	Successful file upload.
9.4	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.
9.5	n files in storage.	User tries to upload a file with an invalid type.	Error message indicating file has not been uploaded.
9.6	n files in storage.	User requests to download a file.	Successful file download.
9.7	n files in storage.	Connection interrupted while download is in progress.	Error message indicating file has not been downloaded.
9.8	n files in storage.	User tries to upload $n > 1$ files.	Error message indicating only one file can be uploaded at a time.
9.9	n files in storage.	User clicks delete file.	File removed.

4.10 Bulletin Board

Test Type: Functional, Dynamic, Automated.
Tools Used: Custom Scripts.
Schedule: Begin testing after the PoC Demo. Complete automated tests by Final Demo April 1.
Team Member Responsible: James Anthony.
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
10.1	No posts on bulletin board.	A post with 0 characters	Empty post is disgarded ["discarded" —DS] and not added to bulletin board.
10.2	No posts on bulletin board.	A post with n characters, where $n > 0$.	Bulletin board is updated with the post of n characters.
10.3	p posts on bulletin board, where $p > 0$.	A post with 0 characters	Empty post is disgarded and not added to bulletin board.
10.4	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.
10.5	p posts on bulletin board, where $p > 0$.	A comment with 0 characters on an existing post p.	Empty comment is disgarded [discarded —DS] and not added to bulletin board.
10.6	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.

5 Non-Functional Tests

[Nonfunctional qualities are evaluated as appropriate. These qualities include usability,performance, and robustness. Quantify results. If these tests are not performed, there absence should be explicitly justified. —CC]

5.1 Look and Feel

To test that the system is attractive and intuitive and appears professional and secure, we will survey ten users to rate the user interface on a scale of 1 to 10, where a 1 means "ugly, unprofessional and would not return to the site", and a 10 means "captivating, professional, and would refer a friend". We will also ask users to provide comments or suggestions for qualitative feedback. The testing schedule will include a test December 7 and February 22. The first test will be used as a baseline. We will do a second test before the Final Demo to see if we improved.

[If you are providing a survey, you should include a copy of it as an appendix. —DS]

5.2 Usability

To test that the system is intuitive to use and navigate, we will ask ten users to complete a set of tasks on the site. Five users will act as landlords, who will create an account, login, send an email invitation to invite users to a house, and then create a post on the discussion board. Five other users will act as tenants who will accept an email invitation, create an account, login, and then create a post on the discussion board. Users will rate their experience on a scale of 1 to 10, where a 1 is "frustrated, could not complete task(s), would not recommend", and a 10 means "user friendly, easy to navigate, would recommend to a friend". We will also ask users to provide comments or suggestions for qualitative feedback. The testing schedule will include a test December 7 and February 22. The first test will be used as a baseline. We will do a second test before the Final Demo to see if we improved.

[Can you more accurately describe what the users will be tasked with doing? —DS]

5.3 Performance

To test the server, we will do a load testing to make sure the server can handle 100 simultaneous requests.

[How? —DS]

5.4 Robustness

To test the security of the system, including file access, failed password attempts, SQL injections, and expired sessions, we will do manual testing.

[Be more descriptive. —DS]

6 Summary of Changes

[Summarize changes made in response to testing. —CC]