# Software Requirements Specifications for Quarters

James Anthony (1135089)
Wenqiang Chen(1155437)
Carolyn Chong (1139105)
Kevin Ly (1144604)

February 29, 2016

# Contents

# List of Figures

# List of Tables

# Revision History

| **Date** | **Comments** |
|---|---|
| October 9, 2015 | Created first draft. |
| January 17, 2016 | Made some revisions based on marking feedback, including Other Stakeholders, Users of the Product, Constraints, Work Partitioning, Product Use Case Diagram, Nonfunctional requirements, and general spelling errors. |
| January 18, 2016 | Fixed Constraints and Ideas for Solutions. |
| February 8-10, 2016 | Updated Naming Conventions and Terminology, Users of the Product, Work Partitioning Table, Functional Requirements, Waiting Room. |

# Template

This document makes use of the Volere Template for all of its organization.

# 1 Project Drivers

## 1.1 The Purpose of the Project

### 1.1.1 Project Background

Communication is the exchange of information between two parties. It is an important aspect of people's everyday lives. With the introduction of the Internet and mobile devices people's capacity to communicate has vastly improved, however, this information has hardly centralized. Landlords and their tenants commonly communicate via email, instant messaging or phone calls, but a response is not always guaranteed or the reply is hard to retrace. This lack of centralized communication may introduce confusion and frustration between the two parties in addition to challenges sharing documents and other important information.

Developing a new software platform to serve as an intermediate between landlords and tenants to act as a centralized hub of information will improve the quality of communication and create a more functional living environment for the occupants.

### 1.1.2 Project Goal

This platform is designed to handle information between landlords and their tenants, as well as between other tenants. It efficiently and systemically handles communication between all parties and gives a detailed status of the household.

## 1.2 The Client, the Customer, and Other Stakeholders

### 1.2.1 The Client

N/A.

### 1.2.2 The Customer

This platform is designed for tenants and landlords. It provides tenants with the ability to communicate efficiently and effectively with each other and their landlord. The customer requires a robust, easy-to-learn platform.

### 1.2.3 Other Stakeholders

Other stakeholders whose input is needed to build the product include the project supervisor to provide guidance and advice on the design and implementation, current software developers to create and test the application and future software developers to improve and build on the existing version.

### 1.3 Users of the Product

There are two key users of the product: tenants and landlords. From herein, the term ”user” encompasses both a tenant and landlord, unless otherwise specified.

Tenants have the following role:

1. Communicate information to other tenants when needed
2. Complete the chores assigned to them
3. Report maintenance issues in the house via the ticketing system
4. Be punctual with payments
5. Ensure the rules outlined for the house are followed
6. 18+ years old
7. Basic web browsing experience required

Landlords have the following role:

1. Communicate information regarding the house to the tenants
2. Respond to tenant questions and inquiries
3. Complete maintenance requests within a timely manner
4. Be available when issues arise
5. 18+ years old
6. Basic web browsing experience required

# 2 Project Constraints

## 2.1 Solution Constraints

**Constraint #:** 1
**Description:** The web application must run on an Intel i3-4430 server running Ubuntu.
**Rationale:** The team of developers already had access to this server.
**Fit Criterion:** The web application works on the server.

## 2.2 Implementation Environment of the Current System

N/A.

## 2.3 Partner or Collaborative Applications

N/A.

## 2.4 Off-the-Shelf Software

N/A.

## 2.5 Anticipated Workplace Environment

- Home: Website must display properly on desktop and laptop computers.
- Mobile: Website must display properly on mobile browsers.

## 2.6 Schedule Constraints

- Proof of Concept Demonstration on November 24, 2015
- Revision 0 Demonstration on February 10, 2016
- Final Demonstration in April 2016

## 2.7 Budget Constraints

N/A.

## 2.8 Enterprise Constraints

N/A.

# 3 Naming Conventions and Terminology

- House: In the context of this project, a house functions as a set which contains one or more users and stores information about the physical property, the users, and content added by those users.
- User: A user is a user of the application. A user is designated as an administrator or member of a house.
- Administrator: The user that creates the house is, by default, the administrator of the house. The administrator of a house is the only member of the house who can change information about the house, upload/delete files, add/delete members, and delete the house.
- Maintenance request: A ticket created by a member of a house to inform another member of the same house of property-related maintenance that needs to be addressed.

# 4 Relevant Facts and Assumptions

## 4.1 Relevant Facts

N/A.

## 4.2 Assumptions

- It is assumed that both landlords and tenants will be capable of using web applications for communication, planning, and payments.

# 5 Functional Requirements

## 5.1 The Scope of the Work

### 5.1.1 The Current Situation

There is currently no existing software platform that attempts to simplify and document communication between landlords and tenants. A web application is needed to serve as a centralized management solution that will benefit both types of users. The web application will include document storage, financial transaction history, a calendar, maintenance ticketing, and a bulletin board.

### 5.1.2 The Context of the Work

See Figure 1.

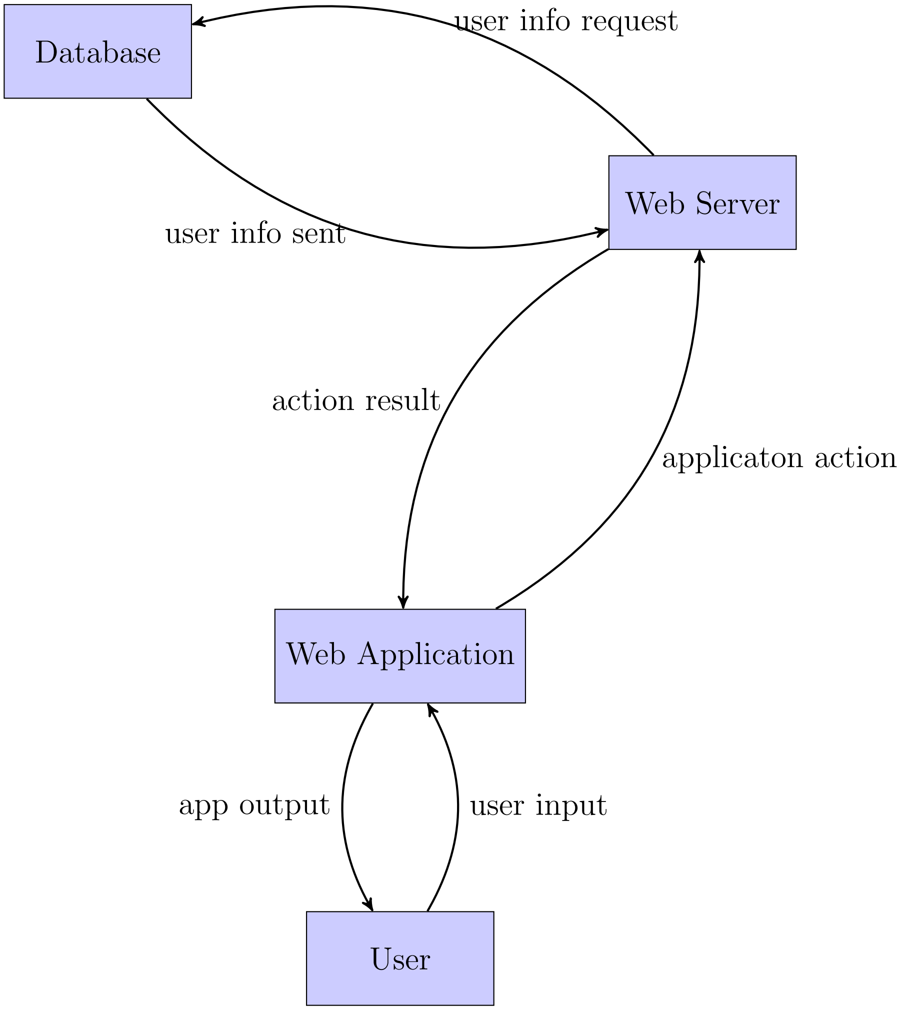

Figure 1: Work Context Diagram

### 5.1.3 Work Partitioning

See Table 2.

| **Event Name** | **Input and Output** | **Summary** |
|---|---|---|
| 1. User creates account. | User email (IN)<br>User password (IN) | User creates an account. The system then adds the account to the database. |
| 2. User deletes account. | User email (IN)<br>User password (IN) | User deletes their account. The system then removes the account from the database. |

| | | |
|---|---|---|
| 3. User logs in. | User email (IN)<br>User password (IN) | User logs in to app. |
| 4. User logs out. | User email (IN) | User logs out of app. |
| 5. User creates house. | House address (IN)<br>Invitation code (OUT) | User inputs the house address into system. The system then saves the house in the database. The user that created the house becomes, by default, the administrator of the house. A unique invitation code is generated that the house creator uses to invite other users to join the house. |
| 6. User joins house. | User email (IN)<br>Invitation code (IN) | User enters invitation code. The system adds the user to the house. |
| 7. User leaves house. | User email (IN)<br>House address (IN) | User selects a house to leave. The system removes the user from the house. |
| 8. User uploads file. | User email (IN)<br>House address (IN)<br>File (IN) | User adds a file to be visible to house. The system then saves the file in the database. |
| 9. User submits maintenance request. | User email (IN)<br>House address (IN)<br>Request (IN)<br>Request (OUT) | User submits a maintenance request to be received by another user of the same house. |
| 10. User updates maintenance request. | User email (IN)<br>House address (IN)<br>Request (IN)<br>Request (OUT) | User marks a maintenance request as complete. |
| 11. User adds chore/event. | User email (IN)<br>House address (IN)<br>Chore/Event (IN)<br>Calendar (OUT) | User submits a chore or event to be displayed on the Calendar. |
| 12. User deletes chore/event. | User email (IN)<br>House address (IN)<br>Chore/Event (IN)<br>Calendar (OUT) | User deletes a chore to be removed from the Calendar. |

| | | |
|---|---|---|
| 13. User adds post. | User email (IN)<br>House address (IN)<br>Post (IN)<br>Bulletin Board (OUT) | User adds post to bulletin board. |
| 14. User comments on post. | User email (IN)<br>House address (IN)<br>Comment (IN)<br>Bulletin Board (OUT) | User comments on post on bulletin board. |
| 15. User deletes post. | User email (IN)<br>House address (IN)<br>Post (IN)<br>Bulletin Board (OUT) | User deletes post from bulletin board. |
| 16. User adds record of financial transaction. | User A email (IN)<br>User B email (IN)<br>House address (IN)<br>Record of transaction (IN)<br>Transaction history (OUT) | User A enters the monetary amount and the name of User B (another user belonging to the same house) with whom they completed an external financial transaction. A record of the transaction is displayed. |

Table 2: Work Partitioning

## 5.2 Business Data Model and Data Dictionary

N/A.

## 5.3 The Scope of the Product

### 5.3.1 Product Boundary

N/A.

## 5.4 Functional and Data Requirements

**Requirement #:** 1 **Event/Use Case:** 1 **Priority:** 5
**Description:** Account registration.
**Rationale:** To individualize a user's experience and ensure security of a user's information and data.
**Fit Criterion:** Only a registered user can use the application.

**Requirement #:** 2 **Event/Use Case:** 2 **Priority:** 2
**Description:** Account removal.
**Rationale:** For users who wish to delete their account from the system.
**Fit Criterion:** Once a user deletes their account they cannot use the application unless they create a new account.

**Requirement #:** 3 **Event/Use Case:** 3 **Priority:** 5
**Description:** User login.
**Rationale:** All users should be able to securely login.
**Fit Criterion:** Upon passing proper credentials, user completes the login process. If user inputs incorrect email/password combination, the user is prompted to try again.

**Requirement #:** 4 **Event/Use Case:** 4 **Priority:** 5
**Description:** User logout.
**Rationale:** All users should be able to securely logout.
**Fit Criterion:** Upon logging out, a user must login to access their account.

**Requirement #:** 6 **Event/Use Case:** 5 **Priority:** 5
**Description:** House creation.
**Rationale:** To allow users to self-organize and communicate.
**Fit Criterion:** A house object is created in the database, with the creator as its default administrator.

**Requirement #:** 7 **Event/Use Case:** 6 **Priority:** 5
**Description:** User joins a house.
**Rationale:** To allow multiple users to join house groups which have already been created.
**Fit Criterion:** The user is added to the list of members associated with the house they are joining.

**Requirement #:** 8 **Event/Use Case:** 7 **Priority:** 5
**Description:** User leaves a house.
**Rationale:** To allow a user that is a member of a house to leave the house group.
**Fit Criterion:** The user is removed from the list of members associated with the house they are leaving.

**Requirement #:** 9 **Event/Use Case:** **Priority:** 5
**Description:** House management section.
**Rationale:** To view: the list of houses a user belongs to, house documents, member profiles, and house details. Also, to allow user to switch house views.
**Fit Criterion:** House information is accurate and user can switch which house to view.

**Requirement #:** 10 **Event/Use Case:** 11 **Priority:** 4
**Description:** Calendar section.
**Rationale:** To allow users to keep track of upcoming chores and events.
**Fit Criterion:** Calendar format should be based off the standardized ICS structure. This platform should support the addition and deletion of an event.

**Requirement #:** 11 **Event/Use Case:** **Priority:** 5
**Description:** Multiple resolution compatibility.
**Rationale:** Users may have different devices with different screen resolutions. The application should be able to support all types.
**Fit Criterion:** Web platform should support mobile screens and desktop screens, including both horizontal and vertical layouts.

# 6 Nonfunctional Requirements

## 6.1 Look and Feel Requirements

### 6.1.1 Appearance Requirements

The interface of the web application shall be attractive and intuitive for a young adult and adult audience.

### 6.1.2 Style Requirements

The web application shall appear professional and secure.

## 6.2 Usability and Humanity Requirements

### 6.2.1 Ease of Use Requirements

The web application shall be used by users with no prior training. A casual user should be able to use the application with the same ease of a frequent user. The utility of the application and its unique advantages shall be apparent upon a first encounter. A test panel of current landlords and their tenants shall be able to successfully create a user account and use the application's functions without guidance within their first encounter.

### 6.2.2 Personalization and Internationalization Requirements

The web application shall be available in the Canadian English language (EN-CA), use Canadian currency (CAD $), ICS Calendar format, and the metric system.

### 6.2.3 Learning Requirements

The web application shall be easy to learn for users 18 years of age or older. The web application shall be constructed so that all of its functionality is apparent upon first encountering it. A brief tour of the web application shall be presented as an option to first time visitors of the site. A test panel of current landlords and their tenants shall be able to successfully create a user account and use the application's functions productively without guidance within their first encounter.

### 6.2.4 Understandability and Politeness Requirements

The web application shall use symbols, icons, and words that users have seen and used before on other web applications.

### 6.2.5 Accessibility Requirements

N/A.

## 6.3 Performance Requirements

### 6.3.1 Speed and Latency Requirements

N/A.

### 6.3.2 Safety-Critical Requirements

N/A.

### 6.3.3 Precision or Accuracy Requirements

The web application shall keep accurate time by working in UTC. All monetary amounts shall be accurate to two decimal places.

### 6.3.4 Reliability and Availability Requirements

The web application shall be available for use 24 hours per day, 365 days per year, with the exception of one hour per week for maintenance and bug fixes.

### 6.3.5 Robustness or Fault-Tolerance Requirements

The web application shall successfully display an error message to the user should an incorrect username/password combination be input, or in the event one of its features crashes.

### 6.3.6 Capacity Requirements

The web application shall cater to 100 simultaneous users during its initial release. Capacity will increase with expansion.

### 6.3.7 Scalability or Extensibility Requirements

The web application shall be capable of expanding to nearby cities within two years of its launch.

### 6.3.8 Longevity Requirements

The web application shall be expected to operate as long as there exists a housing rental market.

## 6.4 Operational and Environmental Requirements

### 6.4.1 Expected Physical Environment

N/A.

### 6.4.2 Requirements for Interfacing with Adjacent Systems

The web application shall work on the last three major releases of the five most popular web browsers (Chrome, Firefox, Internet Explorer, Opera, Safari). The details of the communication standards/protocols will be outlined in the Design Document.

### 6.4.3 Productization Requirements

The web application shall be accessible on the World Wide Web.

### 6.4.4 Release Requirements

The initial release of the web application will be February 10, 2016. The next release will be in April 2016.

## 6.5 Maintainability and Support Requirements

### 6.5.1 Maintenance Requirements

N/A.

### 6.5.2 Supportability Requirements

A user help guide and FAQ section will be made available to all users; new users will be given a tour of the application.

### 6.5.3 Adaptability Requirements

The web application is expected to run on web browsers on mobile phones, tablets and desktop computers.

## 6.6 Security Requirements

### 6.6.1 Access Requirements

Only the user has access to edit their own personal stored information. Only members belonging to the same property can view the property's group and add content to the property's group.

### 6.6.2 Integrity Requirements

The web application shall prevent incorrect data from being introduced and protect itself from unwanted attacks by unauthorized users. The web application shall have a back-up of its stored data on an alternate server. Back-ups will be completed once per day.

### 6.6.3 Privacy Requirements

Data shall be stored securely and be tamper-proof.

### 6.6.4 Audit Requirements

N/A.

### 6.6.5 Immunity Requirements

N/A.

## 6.7 Cultural and Political Requirements

N/A.

## 6.8 Legal Requirements

N/A.

# 7 Project Issues

## 7.1 Open Issues

- Size of the user group is uncertain, therefore a hardware upgrade may be required in the future to accommodate the user.

- Results of usability tests may completely change business model.
- The scope of the project given project deadlines.
- Security and permissions.

## 7.2 Off the Shelf Solutions

### 7.2.1 Ready-Made Products

No ready-made products exist with the same functionalities.

### 7.2.2 Reusable Components

N/A.

### 7.2.3 Products That Can Be Copied

N/A.

## 7.3 New Problems

### 7.3.1 Effects on the Current Environment

N/A.

### 7.3.2 Effects on the Installed Systems

N/A.

### 7.3.3 Potential User Problems

N/A.

### 7.3.4 Limitations in the Anticipated Implementation Environment That May Inhibit the New Product

Old web browsers are not compatible with the web application.

### 7.3.5 Follow-Up Problems

- User abusing the system.
- User uploading sensitive data.

## 7.4 Tasks

### 7.4.1 Project Planning

- Present requirements document to supervisor for feedback.
- Develop prototype for demo.
- Test, refine and develop more features.

### 7.4.2 Planning of the Development Phases

- Design the database together.
- Design a generalized UI.
- Design back-end of the application.
- Split application into different modules and assign one module to each member for completion.

## 7.5 Migration to New Product

N/A.

## 7.6 Risks

- Security issues with user accounts and data.
- Not able to get enough users.
- Certain features may not be compatible with the hardware.
- Project becomes too complicated and not able to meet deadlines.

## 7.7 Costs

- Domain costs and web server costs, if we decide to go live.
- Approximately 7 months of development time.

## 7.8 User Documentation and Training

### 7.8.1 User Documentation Requirements

- A help guide will be included.
- FAQ section.
- Tour of the website is shown for first time user.

### 7.8.2 Training Requirements

No training is required for the user. When they visit for the first time they will be given a tour.

## 7.9 Waiting Room

The next release will include the following features:

- In-app payment.
- Repeating events and colour-coded events on Calendar.
- Separate module for housing advertisements.
- Ability to attach pictures and files in discussion board.
- Notifications either through email or text messages (for urgent events).
- Update to Bootstrap v4.
- House deletion.
- Administrator roles.
- Maintenance ticketing.
- Finance History.
- Bulletin board.
- Document upload.

## 7.10 Ideas for Solutions

- PostgreSQL: An object-relational database management system (ORDMBS) with an emphasis on extensibility and on standards-compliance.
- ExpressJS: A NodeJS web application server framework, designated for building single-page, multi-page, and hybrid web applications.
- NodeJS: An open-source, cross-platform run-time environment for developing server-side web applications.
- Bootstrap: A free and open-source collection of tools for creating websites and web applications.
- FullCalendar: An open-source full-sized JavaScript calendar to display events.