

Software Design Document

CrashPad

Vision Document for CrashPad

© 2020 CrashPad

Revision History

Date	Revision	Description	Author
06/22/2020	0.0	Document Created	Anthony Woods
06/23/2020	0.1	added objects	Austin Parks
06/23/2020	0.2	Added 4.2 - System Architecture	Arya Habibi
06/24/2020	0.3	Added 4.3.1 & 4.3.2 Interface descriptions	Anthony Woods
06/24/2020	0.4	Added additional objects	Arya Habibi
06/26/2020	0.5	Added interfaces	Aaron Elam
06/26/2020	0.6	Fixed formatting and started on section 3.1	Arya Habibi
6/26/2020	0.7	Added UI Mockups Document Formatting	Alex Pettyjohn
6/26/2020	0.8	Added Chat related Objects	Arya Habibi
6/27/2020	0.9	Added Use Case diagrams	Aaron Elam
6/28/2020	0.99	Formatting, Page Layout, Section 4	Alex Pettyjohn
6/28/2020	1.0	Ready to Submit	All

Development Team: Aaron Elam, Alex Pettyjohn, Anthony Woods, Austin Parks, Arya Habibi

Course: CS 3398 Software Engineering, Summer 1, 2020

Table of Contents

1	Introduction	4
1.1	Purpose	4
1.2	Solution Overview	4
1.3	References	4
2	Glossary	5
2.1	Definitions, Acronyms & Abbreviations	5
3	Use Cases	6
3.1	Actors	6
3.2	List of Use Cases	6
3.3	Use Case Diagrams	7
3.4	Use Cases.	10
4	Design Overview	15
4.1	Introduction	15
4.2	System Architecture	15
4.3	System Interfaces	16
4.4	Assumptions and Constraints	23
5	Objects.	24
5.1	Objects.	24
6	Non Functional Requirements.	27
6.1	Usability	27
6.2	Reliability	27
6.3	Performance	28
6.4	Supportability	28
6.5	Other Requirements	28

1 Introduction

1.1 Purpose

The purpose of this document is to build an online system for listing and booking short term rentals. Hosts range from single unit owners to large businesses. Guests include personal travelers, international travelers, and corporate clients scheduling business trips. The proposed platform must provide competitive functionalities while remaining easy to use and reliable. Guests must be able to book and manage their reservations as well as provide feedback to the property owner. Owners must be able to list their units, apply any special conditions they have for renting, and manage a landlord account.

1.2 Solution Overview

The proposed application combines the best features of existing online booking programs with a simple to use user interface and robust features that appeal to each user group. By combining user requested features into a single web application, users can minimize the need to handle many different platforms with individual logins, restrictions, and shortcomings. Property owners (Hosts) will list their availability to a platform that advertises their units for them. These owners need only fill out a form containing the appropriate information to create a listing. Guests will then be able to search these listings by characteristics and availability dates to find the right fit for them, then book their stay using the same application. Payment and agreement to terms will be handled by a standardized method across the web application to remove the concerns for security from both ends of the transaction. Renters and hosts will be put in touch to arrange in person check in and check out processes like acquiring keys, but all other aspects of the transaction will be handled through the web application. After the stay, both the host and renter will be prompted to provide feedback that will be available for future users to review.

1.3 References

Software Vision Document (SVD) template provided by Dr. Palacios for use within his software engineering course. The completed SVD from Project 1 was further developed into the system under design. Dr. Palacios additionally provided a completed Software Design Document (SDD) as a reference for section contents. Definitions of key

terminology drawn from *Engineering Software Products* (Somerville 2015) and lecture materials from CS 3398.

Data on competitors for the proposed software were drawn from annual earnings and customer information reports published by Airbnb and Vrbo. Requirements for web browsers are averaged from Mozilla Firefox, Google Chrome, and Microsoft Edge published software requirements. Standards for web design and implementation drawn from W3C and the Internet Assigned Numbers Authority.

2 Glossary

<u>Term</u>	<u>Definition</u>
Booking:	The process of renting a property.
CSS:	Cascading Style Sheets is a language for styling documents on the web.
GDPR:	The General Data Protection Regulation, a law put in place by the European Union to protect user data.
Guest:	User of the application seeking short-term rental.
Host:	User of the application seeking to rent out the property they own for short-term rental.
HTML:	Hypertext Markup Language is a language for displaying documents on the web.
Short-term rental:	Furnished property that is rented out for a short period of time (usually days to weeks).
Use Case:	The interaction between a role (user or system) and a system to achieve a goal.
W3C:	World Wide Web Consortium, an organization that defines standards for the web.

3 Use Cases

3.1 Use Case Actors

3.1.1 Guest Actor

The Guest Actor is a user who has the ability to create reservations. This account type is the basis for all accounts and any other account types should be considered Guest Accounts with additional flags and permissions added. The Guest Actor can create and manage reservations as well as read and post reviews for a property. The Guest Actor can be seen in multiple use cases such as canceling a reservation or searching a property. Additionally, the Guest Actor can include many different types of users (e.g. Business traveler, Vacationer, or Personal traveler).

3.1.2 Host Actor

The Host Actor is a user who has the permission to create and manage listings. The Host actor starts out with the same permissions as a Guest actor, however, the Host actor has an additional flag enabled which gives the user extra permissions such as the ability to manage properties or create listings. The Host actor can include many different types of users (e.g. Rental Company, Personal property owner).

3.2 List of Use Cases

3.2.1 Message Translation (Server)

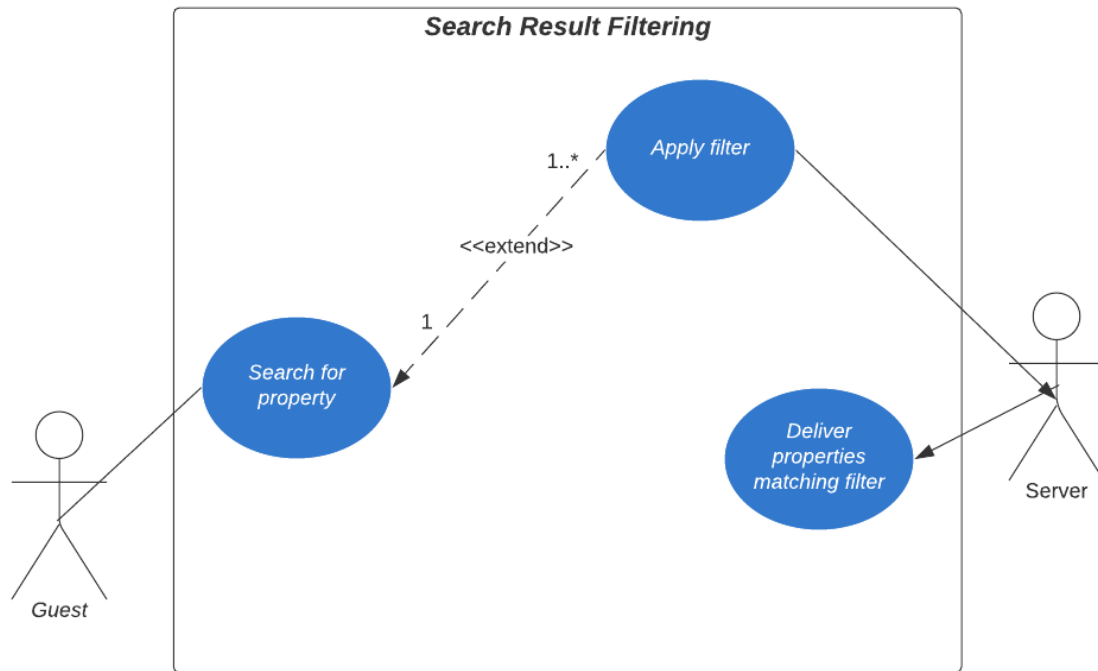
3.2.2 Search Result Filtering (Server)

3.2.3 List a Property (Client)

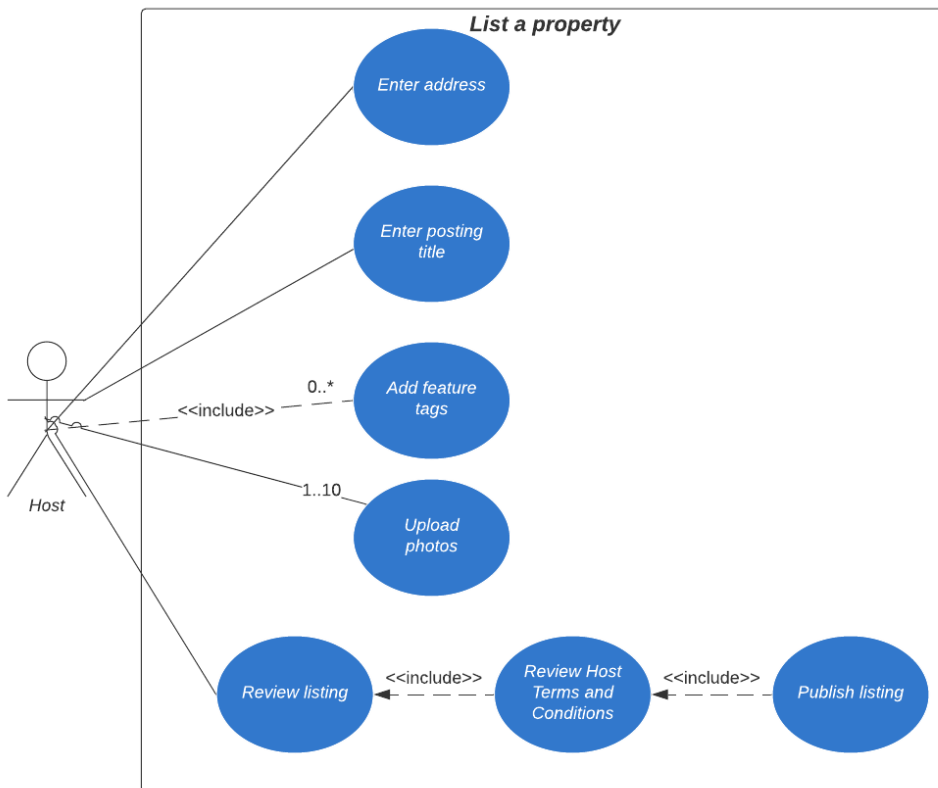
3.2.4 User Submitted Review (Client)

3.2.5 Cancel a Reservation (Client)

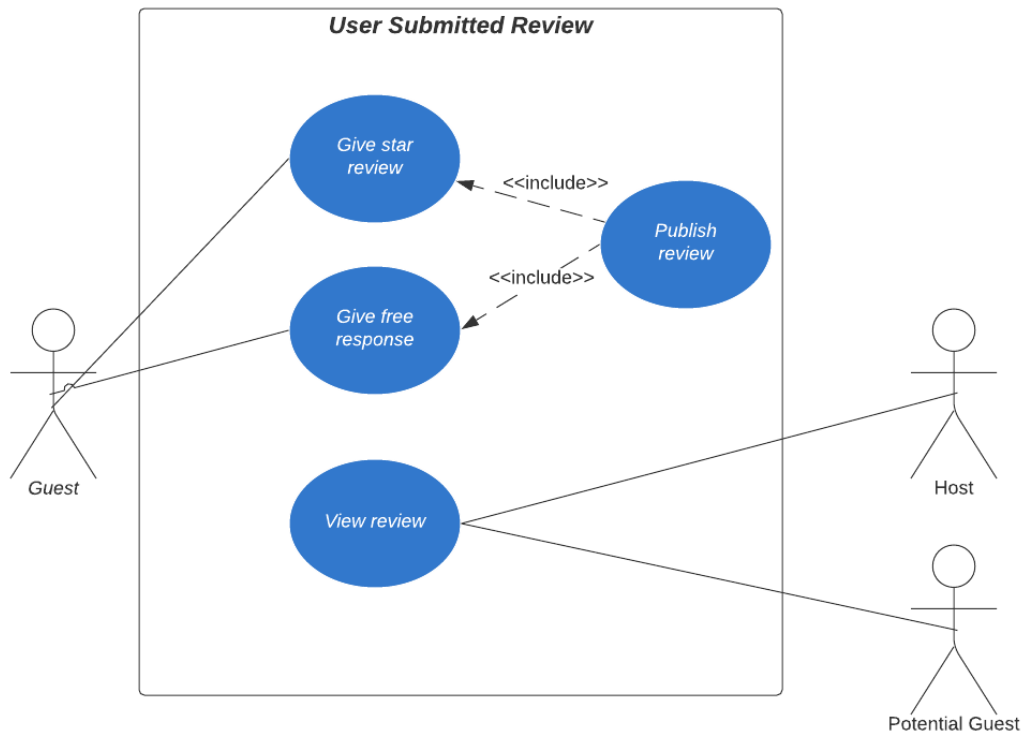
3.3.2 Search Result Filtering



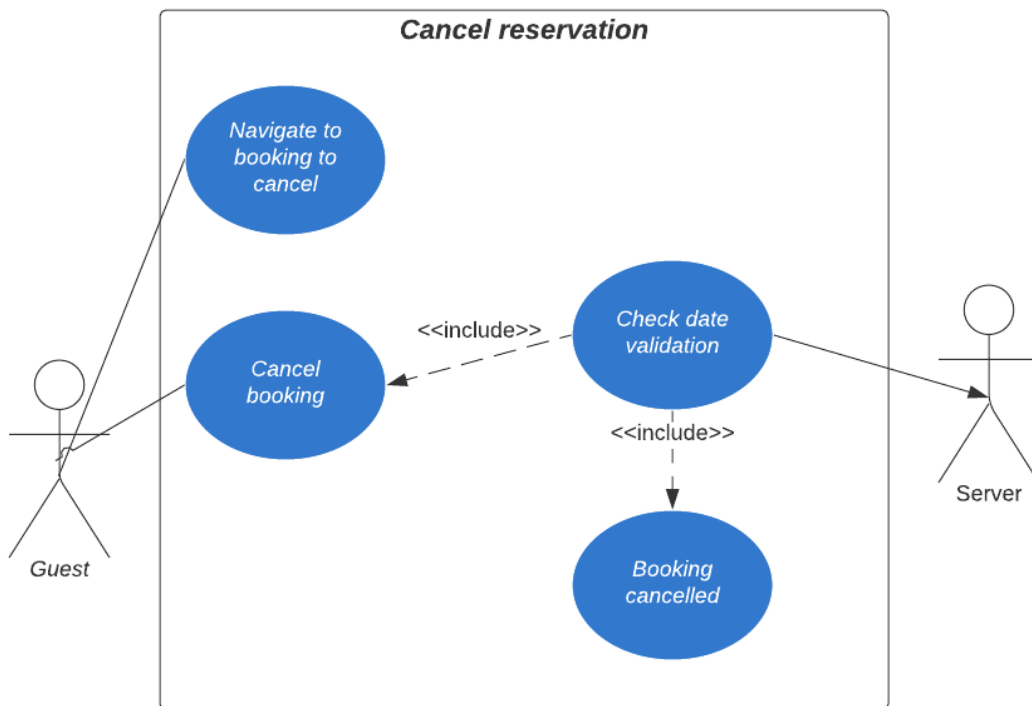
3.3.3 List a Property



3.3.4 User Submitted Review



3.3.5 Cancel a Reservation



3.4 Use Cases

3.4.1 Message Translation

Use Case Name: Message Translation		ID: 0001	Priority: Medium
Primary Actor: Guest or Host	Source: Guest or Host	Use Case Type: Business	Level: User
Interested Stakeholders: End Users, Support			
Brief Description: After successful booking of a rental, the user decides to contact the host of the rental asking a question about an amenity listed. The host sees this message in their language along with a notification that it is translated. The host replies and they are able to sort out the question.			
Goal: Clear communication between users regardless of their preferred language.			
Success Measurement: Message arrives translated with an error rate under 1%			
Precondition: Message sent by registered user Recipient has set a preferred language in their account settings Trigger: User and recipient have set different preferred languages			
Typical Flow of Events: <ol style="list-style-type: none">1. Sender writes message in his/her preferred language2. Message language is detected after sending3. Message is translated4. Message delivered to recipient in preferred language			
Assumptions: Both sender and receiver have preferred languages, would otherwise write in different languages			
Implementation Constraints and Specifications: Preferred language must be maintained when viewing a sent message from either end. Initial senders should see message history in their preferred language, while the initial recipient will see all messages in their own.			

3.4.2 Search Result Filtering

Use Case Name: Search Result Filtering		ID: 0002	Priority: Medium
Primary Actor: Guest	Source: Guest	Use Case Type: Business	Level: User
Interested Stakeholders: End Users			
Brief Description: When looking for a rental, the user is presented with search filters above the listings found in the city the user wants to book in. Amenities and needs that have been found to be the most important are featured. The user applies two of the filters and is presented with properties that have been listed with those attributes. If a property does not have them, the user can report the property.			
Goal: Allow users to sort search results using descriptive criteria.			
Success Measurement: Listings are filtered accurately and by predefined criteria when keyword search is inadequate			
Precondition: Host must tag their listing with certain searchable traits (e.g. “pet friendly”) Trigger: During a search, prospective guest selects to filter by the given search condition			
Typical Flow of Events: <ol style="list-style-type: none">1. Guest initiates a property search2. Specific options to narrow results are presented3. Guest selects 1 or more conditions to narrow results4. Result list changes as more conditions are selected5. Final results are narrowed until Guest is happy with results			
Assumptions: The user desires some specific feature that is not common to all listings			
Implementation Constraints and Specifications: Search criteria must be specific enough to narrow listings. Criteria must be exhaustive in all categories. Options should be mutually exclusive such that no listing belongs to multiple categories in the same option (e.g. Cannot belong to the “under \$100” category and the “\$100-200” category at the same time)			

3.4.3 List a Property

Use Case Name: List a Property		ID: 0003	Priority: Medium
Primary Actor: Host	Source: Host	Use Case Type: Business	Level: User
Interested Stakeholders: End Users, Support			
Brief Description: The host starts creating a listing. During listing, there is a list of other properties in the area, what they offer, and their prices. After this initial research, a checklist of amenities is presented and the user selects what they can offer. Afterwards, the user can set a price and list the property.			
Goal: Host users list a property as available for rent and include relevant details/photos.			
Success Measurement: Generated listing is satisfactory to the host and contains all relevant listing information			
Precondition: Host is the owner of one or more properties			
Trigger: Host creates account with the intent to list a property			
Typical Flow of Events: <ol style="list-style-type: none"> 1. Host is prompted to enter basic details 2. Host is prompted to tag features of the property 3. Host uploads pictures of the property 4. Host agrees to terms and conditions 5. Host reviews listing and publishes 			
Assumptions: Host has the appropriate permissions to rent their unit. Host lives within the service area and is not barred from short term rentals under local law.			
Implementation Constraints and Specifications: Listings should be checked for duplication to prevent “spam” on the platform. Verification of property ownership must be conducted before listing is made public.			

3.4.4 User Submitted Review

Use Case Name: User Submitted Review		ID: 0004	Priority: Medium
Primary Actor: Guest	Source: Guest or Host	Use Case Type: Business	Level: User
Interested Stakeholders: End Users, Support			
Brief Description: A guest stayed at a rental and decided to leave feedback for other guests to see. The user goes to the app, looks at the recent rentals, and clicks on the rental they want to review. The user then writes a review that is published for others to see. Users searching property listings can then view the published review.			
Goal: Users are prompted to provide feedback on their stay that is made available to future renters.			
Success Measurement: High rate of user reviews after a stay. Reviews are visible within a reasonable time after stay.			
Precondition: User has booked and completed a stay on the platform			
Trigger: After payment is complete, an email reminder to review is sent within 48 hours.			
Typical Flow of Events: <ol style="list-style-type: none">1. Guest stays at a property listed on the platform2. Guest is prompted to review their stay3. Guest responds to questions ranking aspects of their stay4. Guest completes a free response review where they may make specific comments5. Review is published and scores averaged with overall reviews.			
Assumptions: Users have an underlying desire to provide feedback and address both positive and negative aspects of their stay. Feedback should be constructive and not used as an alternative to normal support channels.			
Implementation Constraints and Specifications: Users may only review properties at which they have stayed, and only within a reasonable time period following the stay. Users with serious issues with their stay (severely negative reviews) should be directed to a support page.			

3.4.5 Cancel a Reservation

Use Case Name: Cancelling a Reservation		ID: 0005	Priority: Medium
Primary Actor: Guest	Source: Guest	Use Case Type: Business	Level: User
Interested Stakeholders: End Users, Support			
Brief Description: A guest decides they do not want to keep their reservation. They may cancel the reservation and receive a refund minus a small fee.			
Goal: Allow a guest to easily cancel any reservation and feel satisfied to return to the site to book again.			
Success Measurement: Low number of complaints to support that guests would like to cancel.			
Precondition: User has booked but has not completed a stay at the reservation. The cancellation must be performed 5 or more days prior to the reservation date. Trigger: After booking a reservation, the option to cancel is presented any time the guest views it.			
Typical Flow of Events: <ol style="list-style-type: none">1. Guest books a reservation.2. The guest wishes to cancel a reservation they have booked.3. Platform either notifies the guest that they are ineligible for a refund OR they are given information about fees involved with cancelling.4. Guest completes or disengages from the cancellation.5. The guest is refunded the amount of the reservation minus a fee.6. Host is notified of cancellation from the date.			
Assumptions: Assuming the guest will be notified and remember the 5 day cancellation rule.			
Implementation Constraints and Specifications: Payment processors will be outside of our control. Ensuring that guests and hosts are compensated correctly will be difficult.			

4 Design Overview

4.1 Introduction

The system under design will be constructed using a Client-Server architecture. Individual users will run the client application within a web browser while the bulk of the application's work is completed on a remote server. The client and server applications are detailed in this section. The client web application will serve as the primary user interface. This section further presents mockups of proposed interfaces based on expected use cases.

4.2 System Architecture

4.2.1 Client-Server Model

The Application as a whole implements a Client-Server Model as its main method of operation. The Client-Server Model allows the webpage / client application to be lightweight and requires minimal processing power. Most of the data processing is sent through requests to the main server which will then process those requests. To do this, client and server will communicate using API calls over TLS/SSL to ensure that the security and integrity of the requests are being maintained.

4.2.3 Client Application

The client application will be a web based application that pulls all data from the main application server. This extraction of data from the server will be done using API requests from the guest's browser. Additionally, this same methodology will be used to upload information into storage - such as in the case of uploading a new listing or reviewing a property. The client application requests input from the user and transmits those requests to a central server to ensure that bookings are processed and added to the database. The booking process must be transactional, failing if any component of the process is unsuccessful.

4.2.3 Server Application

The server application will serve as a broker between the client application and the data that is being accessed or uploaded. The server itself will be listening for API calls from the client application and then execute the specific command being requested.

Additionally, the server will handle authentication using API calls as well through the OAuth2 Standard. The server must additionally synchronize inputs from multiple clients to ensure that a property cannot become double booked.

4.3 System Interfaces

4.3.1 List a Property Interface

The property listing interface prompts a host user to enter the necessary information to generate their property listing. Each property is connected to a data object from which the property interface can be generated when a listing is clicked. This page prompts each field to be filled and conducts appropriate checking on entered data. When the page is complete, the system checks the entered information and location to ensure that the listing is allowed in the specified area and suggests a nightly rate based on comparable properties. The host is then shown the completed listing page to confirm that information is accurate and that the listing meets their standards.

Browser Header/Address Bar

Property Title: _____
Property Address: _____

Number of Bedrooms: _____
Number of Bathrooms: _____

Select features:
☐ Non Smoking
☐ Washer/Dryer
☐ First Floor
☐ Elevator
☐ Pet Friendly

Enter Property Description: _____







Click to add photos

Suggested Nightly Rate: \$XX.XX
Or
Enter Custom Rate: _____
[Click Here to add Special Offers](#)

Page 17 of 30








4.3.2 Search Interface

Our presented search feature should prove to separate our service from current and future competitors. Users begin by entering a search term and location, then narrow listings using advanced search filters. Users with special needs or requests can filter property listings to view only those of interest. Users will be shown key details in the form of searchable tags. These tags provide valuable information on what amenities are offered at the property. The search screen also displays a photo of the property and its nightly rate for easy browsing.

Browser Header/Address Bar			
Advanced Search Filters: Property Type: <ul style="list-style-type: none"><input type="checkbox"/> Condo<input type="checkbox"/> Apartment<input type="checkbox"/> Townhome<input type="checkbox"/> Single Family<input type="checkbox"/> Multi Family Bedrooms: <ul style="list-style-type: none"><input type="checkbox"/> 1<input type="checkbox"/> 2<input type="checkbox"/> 3<input type="checkbox"/> 4+ Bathrooms: <ul style="list-style-type: none"><input type="checkbox"/> 1<input type="checkbox"/> 2<input type="checkbox"/> 3+ Features: <ul style="list-style-type: none"><input type="checkbox"/> Non-Smoking<input type="checkbox"/> Washer/Dryer<input type="checkbox"/> First Floor....	Search: Enter Search Text Here Near: Enter City		
			
	PROPERTY TITLE \$XX.XX/Night	PROPERTY TITLE \$XX.XX/Night	PROPERTY TITLE \$XX.XX/Night
			
	PROPERTY TITLE \$XX.XX/Night	PROPERTY TITLE \$XX.XX/Night	PROPERTY TITLE \$XX.XX/Night
	PROPERTY TITLE \$XX.XX/Night	PROPERTY TITLE \$XX.XX/Night	PROPERTY TITLE \$XX.XX/Night

4.3.3 View a Property Interface

The property interface is designed to greet the user with all pertinent information needed to make a rental decision. Each listing includes key details of the property, photos, and user submitted reviews. The details of the property are properly highlighted with bedrooms and bathrooms being the most important. Then, features, price, and a description given by the Host. Below the features section is a block for reviews given by previous guests, each of which includes a rating and verification that the review is from a user who stayed at the property.

Browser Header/Address Bar					
<div>EXAMPLE PROPERTY TITLE</div>					
<div>Bedrooms: X Bathrooms: X</div> <div>Property Features: e.g. Non-Smoking, First Floor, Pet Friendly, Washer/Dryer Access</div> <div>Property Description: Host description of the property goes here. Limited character count for easy reading</div>	<div></div>				
<div>Reviews ★★★★★</div> <table border="1"><tbody><tr><td></td><td><div>Username Stayed on xx/xx/xxxx</div><div>Review Text Here</div></td></tr><tr><td></td><td></td></tr></tbody></table>		<div>Username Stayed on xx/xx/xxxx</div> <div>Review Text Here</div>			<div>Nightly Rate: \$12.34</div> <div>Special Pricing Here</div>
	<div>Username Stayed on xx/xx/xxxx</div> <div>Review Text Here</div>				

4.3.4 Messaging Interface

The messaging interface provides an avenue for clear communication. Each conversation is connected to a booking. This allows the messaging window to provide more information about who the user is talking to. The example interface shows a conversation between a guest and their future host, as denoted by the description at the top of the window. The conversation will appear translated to each user's preferred language in their client application. Additional features, such as delivery receipts, typing notifications, and spell checking are added for user convenience.

John Smith
Future Host at Example Property for your stay on xx/xx/xxxx

Example Translated Message.
Appears in receiver's preferred language here, untranslated for sender

Received 10:20am

Example untranslated message.
Appears in sender's language here but translated to receiver

Delivered 10:30am

.... John Smith is typing

Type message here....

View Original

Send

4.3.5 Reservation Management Interface

Provides all information of the Guest's rental in one spot. Here, the user can navigate to the messaging system or booking system. It is also useful because it is cached so that users with no internet connection will still be able to view pertinent information and contact information about the property.

Your booking at Example Property on 06/20/2020	
Booking Dates	06/20 to 06/23
Check in:	after 12pm on 06/20
Check out:	before 11am on 6/23
Nightly Rate:	\$XX.XX
Deposit:	\$XX.XX
Tax:	\$XX.XX
Total:	\$XX.XX
Property Details	Property Features:
1 Bedroom	Washer/Dryer
1 Bathroom	First Floor
Limit 2 guests	Pet Friendly
Contact Host	Modify Booking

4.3.6 Review Submission Interface

After a Guest has completed their stay, the proposed system will send them a notification to review the property. Along with a star rating, the Guest can leave a written review of their stay and attach photos they deem to be fitting of the property experience. Each review will be posted publicly, but users may opt out of their username/profile photo being included in the review page. Reviews may be submitted only by users who have stayed at a property, and each review will include the date of a stay.

Reviewing your stay at Example Property on xx/xx/xxxx

☐ Include your username
☐ Include your profile picture

Enter Star Rating:

★

★

★

☆

☆

Enter Review Text here

(xxx/255)

Attach Photos

Submit Review

4.4 Assumptions and Constraints

4.4.1 List of Assumptions

This application is to be a web based application in which a software download is not required for the use of this product. Guests are assumed to be of legal renting age in both their home jurisdiction and the renting jurisdiction. Violations of these local laws fall on the guest as they agree they are of legal age during account creation. Usage of the system under design requires a stable internet connection and a computer. These factors are common enough that any prospective user can be assumed to have access when necessary. Host users are assumed to be rightful owners/agents of owners with authority to rent the property. These users affirm their right to rent the property during account creation.

4.4.2 List of Constraints

The system will be dependent on proper procedure in place to ensure that the host is the rightful owner/an authorized leasing agent for the property. This authentication is crucial to ensuring that the application functions legally. This system must store property data in a database separate from the web server. Storing this data presents legal challenges with user privacy, particularly in regions covered by GDPR. Certain jurisdictions have made short term renting illegal, or regulated the business by requiring permits. The proposed system cannot operate within jurisdictions where short term rentals are prohibited. Addresses in these areas must be blocked from posting. Where permits are required, the permits must be submitted for review before a host can list a property within the area.

5 Object Descriptions

5.1 Objects

user
<ul style="list-style-type: none">- userId- passwordHash- paymentAccounts- depositAccount- photo- ratings- isHost
<ul style="list-style-type: none">+ getUserId()+ setPasswordHash()+ getPhoto()+ setPhoto()+ getRating()+ addRating()+ avgRating()+ setHost()

Property
<ul style="list-style-type: none">- ownerId- photos- description- address- state- city- zipCode- ratings
<ul style="list-style-type: none">+ getOwnerId()+ getDescription()+ setDescription()+ getAddress+ getState()+ getCity()

+ getZipCode() + addPhoto() + rmPhoto() + getRatings() + addRating() + avgRating()

Rating
- ratingId - title - description - poster - linkingObjId - rating - photos
+ getRatingId() + getTitle() + setTitle() + getDescription() + setDescription() + addPhoto() + rmPhoto() + getLinkingObjId()

Photo
- photoId - linkingObjId - data
+ getPhotoId() + getLinkingObjId() + getData() + rmPhoto()

Booking
<ul style="list-style-type: none"> - linkingGuest - linkingHost - linkingProperty - creationDate - bookingStartDate - bookingEndDate - rating
<ul style="list-style-type: none"> + getGuestId() + getHostId() + getProperty() + setBookingStartDate() + setBookingEndDate() + getBookingStartDate() + getBookingEndDate() + setRating()

Conversation
<ul style="list-style-type: none"> - messageContainer - conversationGuest - conversationHost
<ul style="list-style-type: none"> - getGuest() - getHost() - getMessageById()

Message
<ul style="list-style-type: none"> - messageData - linkingSender - linkingRecipients - sendTime
<ul style="list-style-type: none"> + getTime() + getMessageData() + getSender() + getRecipient()

6 Nonfunctional Requirements

6.1 Usability

The overall user interface of the program will be designed to remain functional and visually appealing to users. Interfaces for site features will be based on industry best practices and seek to feel similar to other applications users likely encounter. This familiar feeling will ensure user comfort and decrease time spent learning the website features. Usability will be assessed through focused user testing with groups representing different types of users (e.g. potential guests, property owners). Feedback from these groups will be the basis for user interface changes. Where possible, features will be implemented to be customized to fit user preferences. Additionally, the site will be made accessible to those with hearing or vision concerns by providing proper alternatives to sound or video based features.

6.2 Reliability

Overall reliability of the system is crucial to maintaining both individual and commercial users. Failure to correctly process user requests may result in their discontinued use of the platforms and leave a lasting negative impact. To address reliability concerns, a number of tests shall be devised and used as acceptance criteria for all revisions/iterations of the software. While these test cases cannot be exhaustive, they will provide a foundation and ensure that core features function in all new versions. The web-based nature of the program allows for revisions to be made without requiring updates on user's machines. This allows for regular bug fixes or improvements to be deployed addressing issues that make it past the test cases. When such issues arise, new test cases shall be created to test that these problems do not reoccur in future versions. The development team understands that no software can be perfect, and will continue to refine and improve the product well after release.

6.3 Performance

Given the variety of alternative renting platforms on the web, providing timely service to users will be required to stay competitive. Performance of web based applications is limited to some degree by the speed of the user's internet connection. To provide the best experience for all users, this application will handle the majority of required operations using web servers. Little processing load will be placed on the users machine. Overall response time will be assessed by time to serve each user request, with the caveat that some users will experience high delay caused by the network rather than by slow response.

6.4 Supportability

The proposed application will be designed with a long life in mind. Its dependence on existing, long lived, and well supported formats such as HTML^[6] and CSS^[7] reduce the likelihood of a shortening of product life due to changes in technology. In the event of changes to these underlying systems, including a major update that would cause deprecation of some features, a long overlap window is generally provided. These windows often last years, providing ample opportunity for the development of alternatives. The user databases are expected to continue to grow over time, with the hope of large growth in user population. Maintaining this information will be a priority for product support after deployment.

6.5 Other Requirements

6.5.1 Applicable Standards

The proposed software will be developed in accordance with the W3C guidelines for web pages, including adherence to the following specific recommendations:

1. The web application shall run using HTML and/or CSS to ensure portability to the wide variety of available browsers and systems that customers may use.
2. Images shall be formatted as either Portable Network Graphics (.PNG) or Scalable Vector Graphics (SVG).
3. Images shall include alternative text to better serve those with limited vision. All site functionalities shall be accessible without input from a mouse.
4. A system shall be implemented to honor "do not track" requests sent by users.

Additionally, and in accordance with the standards set forth by the Internet Assigned Numbers Authority (IANA), the associated domain names and IP addresses will be registered to avoid overlap/conflict with other registered web pages. The IANA time zone database will be used to manage differences in time zones between guests, owners, and the web servers.

6.5.2 System Requirements

As a web-based application, users will require a web browser and a stable internet connection to access the proposed software. Minimum system requirements will vary greatly between browsers (e.g. Google Chrome, Firefox, Safari). The following requirements are an aggregation of those listed in most common browsers:

Operating System:	Windows 7 or later, Mac OS X ‘Yosemite’ or later, or a 64 bit Linux distribution such as Ubuntu 14+ or Debian 8+
Memory:	2GB
Processor:	Intel Pentium 4+ or equivalent (SSE2 or ARM64 support) or any x86-64 CPU for Mac users
Data Storage:	200 MB

6.5.3 Licensing, Security, and Installation

The proposed software will be developed using open source libraries and other software except in cases where no viable open source option is available. As such, no licenses must be obtained to enable development of the application. None of the currently proposed systems depend on a library or software with a licensing requirement.

The following security measures will ensure the protection of sensitive data collected and managed by the proposed application. These data types include the handling of user account information including usernames, passwords, and payment information. Communication between the user, provider, and any intermediaries shall be authenticated to ensure that all parties are correctly identified. These communications will be further checked to ensure that only authorized actions are performed and that attempts to impersonate users with higher level permissions are caught before accessing sensitive data. Data being transmitted will use a secure protocol to prevent tampering and will be further checked for integrity. Where data storage is necessary, all data shall be encrypted using one or more methodologies to make access difficult by unauthorized parties.

No installation of the product is required by the end user as most web browsers include the necessary functionalities to run an HTML based web-application. If the user's browser does not support a current version of HTML, the user will be prompted to update or access the website from a different browser.