Requirements Specification Document

HumbleAbode

ABC Digital Technologies

March 13th, 2022

Table of Contents

List of Tables

List of Figures

Revision History

Name	Date	Reason for Changes	Version
Mark Stoer, Kruti Wani, Brendan Wadey, Elisabeth Klassen, Nanami Momi	Feb. 13, 2022	RD Version 1.0 Final Drafting and Release	0.1
Elisabeth Klassen	Feb. 17, 2022	Updating Functional Requirements	0.2
Mark Stoer, Nanami Momi, Elisabeth Klassen	Feb. 21, 2022	Creation of Use Cases	0.3
Kruti Wani, Mark Stoer	Feb. 22, 2022	Creation of Use Cases	0.4
Brendan Wadey, Nanami Momi	Feb. 23, 2022	Creation of Use Cases, Further Refinement of Use Cases Sections	0.5
Kruti Wani, Nanami Momi, Brendan Wadey	Feb. 24, 2022	Further Editing	0.6
Nanami Momi, Mark Stoer, Brendan Wadey, Kruti Wani, Elisabeth Klassen	Feb. 25, 2022	Final Drafting	0.7
Nanami Momi, Mark Stoer	Feb. 26, 2022	Final Copy	1.0
Mark Stoer	Mar. 7, 2022	Document Versioning Updates	1.1
Elisabeth Klassen	Mar. 10, 2022	Data Dictionary	1.2
Nanami Momi, Mark Stoer, Brendan Wadey	Mar. 11, 2022	ER Diagram, DFD Level 0, DFD Level 1	1.3
Kruti Wani	Mar. 12, 2022	Updating feedback, DFD Level 2	1.4
Kruti Wani, Nanami Momi, Mark Stoer, Elisabeth Klassen Brendan Wadey	Mar. 13, 2022	Final Copy	2.0

1 Introduction

1.1 Purpose

This requirement specification document is for the new extension of Airbnb Inc.’s (Airbnb) web and mobile based rental platform. The extension will be targeted at post-secondary students and rental hosts. Airbnb currently provides a general rental platform where landlords and potential renters can make deals. The new extension, called HumbleAbode, will provide a tailored interface for potential landlords of post-secondary students and post-secondary students. In this document, the software requirements of the first release, release number 2.0, are detailed. This document describes the functions, desired system qualities, interactions with external systems, and constraints of the new HumbleAbode platform. This document is intended for the client organization, HumbleAbode, and the project managers at ABC Digital Technologies’s contracting division.

1.2 Project Scope

The project scope covers a rental platform targeted at the lodging of post-secondary students. This rental platform will include website and mobile interfaces that provide a rental marketplace, forum, payment process, and application process for renting while integrating with Airbnb’s account, messaging and rental marketplace systems. The rental platform will not include handling the legalities around rental contracts between landlords and renters.

The project will address the pain points for landlords and post-secondary students caused by the unique rental circumstances of post-secondary students.

1.3 Glossary of Terms

User	An entity that uses the Airbnb services through signing up for an account.
Verification	A user is said to be verified to use HumbleAbode if they submit their Letter of Enrollment or Proof of Ownership and get approved.
Student	A user who has verified themselves as a post-secondary student through verification services, enabling them access to HumbleAbode services.
Host	A user who is renting out property, and has verified themselves as a property-owner offering rooms for rent through verification services, enabling them access to HumbleAbode services.
Listing	A rental property listed on either Airbnb or HumbleAbode marketplaces.
Review	A feature that lets students who have rented the listing write feedback.

1.4 References

A description of AirCover by Airbnb:
https://www.Airbnb.ca/aircover

Which internet browsers work best for Airbnb by Airbnb:
https://www.Airbnb.ca/help/article/446/which-internet-browsers-work-best-on-Airbnb

A Deep Dive into Airbnb's Server-Driven UI System by Ryan Brooks:
https://medium.com/Airbnb-engineering/a-deep-dive-into-Airbnbs-server-driven-ui-system-842244c5f5

Airbnb's tech stack provided by HumbleAbode:
https://stackshare.io/Airbnb/Airbnb

Airbnb's Identification help by Airbnb
https://www.airbnb.ca/help/article/1237/verifying-your-identity

1.5 Overview

This document details the requirement for the product in the following manner

- Section 2: Overall Description
    - This section includes the product perspective, features, user classes and characteristics, operating environment, design implementation, constraints, assumptions, and dependencies.
- Section 3: System Features
    - This section includes the system features, their functional requirements, and use cases.
- Section 4: External Interface Requirements
    - This section includes the use case diagram and requirements around external user, hardware, software and communication interfaces.
- Section 5: Other Non-Functional Requirements
    - This section includes the performance, safety and security requirements, and the software quality attributes
- Section 6: Other Requirements
    - This section includes the legal requirements.
- Section 7: Data Flow Diagrams
    - This section includes the level 0, 1 and 2 data flow diagrams.
- Section 8: Entity-Relationship Diagram
    - This section includes the entity-relationship diagram and data dictionary.
- Appendix A
    - This appendix includes the list of the open requirements issues that remain to be resolved.
- Appendix B
    - This appendix includes the brief explanations of the various names given that are not unique to this document.
- Appendix C
    - This appendix includes the additional relevant details about Airbnb’s current infrastructure.

2 Overall Description

2.1 Product Perspective

HumbleAbode is a follow-on member of the Airbnb, Inc. line of products. HumbleAbode will exist as an extension of the existing Airbnb marketplace, building on and utilizing much of the existing infrastructure of the Airbnb marketplace.

There are three major features shared between the Airbnb and HumbleAbode marketplaces:

- Airbnb's existing user interfaces, including
  - Website, app, and direct messaging interfaces
- Airbnb's existing databases and servers, including
  - User profile, message history and rental listing storage
- Airbnb's AirCover insurance service

The Airbnb marketplace shares many requirements with those defined with the RFP, such as a simplified user experience, verification via government issued ID, and accurate insurance coverage. By sharing the above features between Airbnb and HumbleAbode, HumbleAbode will be able to satisfy the same requirements in the same way. HumbleAbode will be designed as an unobtrusive add-on with the express intention to not hinder the requirements of Airbnb.

2.2 Product Features

There are several features that HumbleAbode will provide:

- A verification service for users to become classified as students
- A verification service for users to be enabled to list properties on the HumbleAbode marketplace
- A rental marketplace exclusive to students
  - Including listing search UI and pages for individual listings, similar to Airbnb.
- Student quality of life features including listings organized in terms of school terms, and filtering options
- HumbleAbode listing service for hosts, which allows them to seamlessly enable/disable a listing from appearing on the Airbnb marketplace, the HumbleAbode marketplace, or both
- Host quality of life features that include guaranteed payments at the end of every month, and an insurance coverage notification system

2.3 User Classes and Characteristics

HumbleAbode will focus exclusively on two user classes.

Students are the user class expected to be the majority of users of HumbleAbode. They are classified as post-secondary students searching for a place to rent while continuing their studies. Every student has a regular Airbnb user profile that they have verified as belonging to a student via document upload to enter the HumbleAbode marketplace.

Hosts are the second most frequent user class for HumbleAbode. As land is expensive and houses can have multiple bedrooms, it is unlikely the HumbleAbode marketplace would have more hosts than students. Hosts seek tenants to rent to, and have verified themselves as a host by providing proof of identity and proof of ownership of the property, in order to list their rental on the HumbleAbode marketplace. Their presence within the HumbleAbode marketplace rather than the regular Airbnb marketplace will ideally be because of the added features and benefits of listing in the HumbleAbode marketplace. The features afforded to new hosts are identical to existing hosts, and as such new hosts will not be considered as a separate user class from existing hosts.

2.4 Operating Environment

As HumbleAbode will be added onto the existing Airbnb website, HumbleAbode software must coexist with the current Airbnb website’s architecture. Airbnb is not hosted on in-house servers and databases, instead, a majority of the infrastructure is held by Amazon Web Services, including cloud storage, hosting and databases. As such, HumbleAbode must be on the same cloud-based infrastructure. Additionally, HumbleAbode must be developed using the same software tools (outlined in Section 2.5) as Airbnb in order to coexist with the existing website.

2.5 Design and Implementation Constraints

HumbleAbode software will run on the same web servers as Airbnb, which use NGINX. HumbleAbode will exist within the same website as Airbnb, meaning it will be developed via Rails on the Ghost Platform, using React to develop UI and Java to serve as underlying structure. Additionally, HumbleAbode will share Airbnb’s databases, which use MySQL.

For more detailed information on each named software tool, see Appendix B: Software Tool Definitions.

2.6 Assumptions and dependencies

HumbleAbode is solely dependent upon the existing digital infrastructure of the Airbnb marketplace. This includes the existing Amazon cloud services that Airbnb uses, which are Amazon RDS, Amazon EC2, and Amazon S3. More information on these can be found in Appendix B: Software Tool Definitions.

It is assumed that as an add-on to the regular Airbnb website, Airbnb is responsible for management and maintenance of the servers and databases which are to be shared with HumbleAbode. If HumbleAbode requires more resources than the current Airbnb architecture can provide, it is assumed that it is not within the scope of this project to identify and implement possible additional servers and databases. Airbnb will secure and allocate additional resources as needed.

It is assumed that HumbleAbode and its services fall under the legal responsibilities of Airbnb Inc. All legal claims made by students or hosts will be dealt with by Airbnb Inc., in a similar manner as if the claims had been made under the Airbnb marketplace.

3 System Features

3.1 Listing Search

3.1.1 Description and Priority

A student must be able to browse the portal for listings. This is a high-priority feature.

3.1.2 Functional Requirements

REQ-1-1: A student must be able to access a search page for HumbleAbode listings.

REQ-1-2: A student searching for HumbleAbode listings must be able to filter by the following criteria:

1. Location
2. Price
3. Included utilities and amenities
4. Square footage
5. Accessibility
6. Number of roommates
7. Number of occupants other than roommates
8. Number of open rooms
9. Pet policy
10. HumbleAbode exclusivity

REQ-1-3: When a student clicks on a HumbleAbode listing on the search page, the student must be redirected to the listing's page.

3.1.3 Use Cases

UC-1-1	Search Listings
Actor(s)	Primary: Student
Description	The student searches HumbleAbode for a place to live and finds a location to apply for.
Secondary Use Cases	N/A
Precondition(s)	PRE-1: Student is logged in.
Postcondition(s)	POST-1: Student has been redirected to the listing's page.

Normal Flow	1. Student accesses the HumbleAbode search page. 2. Student searches based on a set of filters that are a subset of the available filters (listed in REQ-1-2). 3. Student finds a location and clicks on the preview. 4. Student is redirected to the listing's proper page.

3.2 Listing Pages

3.2.1 Description and Priority

A HumbleAbode listing must have its own page providing all information about it. This is a high-priority feature.

3.2.2 Functional Requirements

REQ-2-1: A student must be able to access a HumbleAbode listing page, which provides the following information:

1. Approximate location
2. Photos
3. Contact information
4. Price
5. Description
6. Included utilities and amenities
7. Square footage
8. Accessibility
9. Number of people currently living in the listing
10. Number of available rooms in the listing
11. Pet policy
12. HumbleAbode exclusivity marker (if applicable)
13. Host name and photo
14. Previous rating and reviews

REQ-2-2: From a HumbleAbode listing page, a student must be able to apply to rent that listing.

3.2.3 Use Cases

UC-2-1	Access HumbleAbode Listing Page
Actor(s)	Primary: Student
Description	A student accesses a HumbleAbode listing page.

Secondary Use Cases	UC-1-1: Search Listings
Precondition(s)	PRE-1: Student searches for listings on HumbleAbode.
Postcondition(s)	POST-1: Student accesses a page with detailed information of the listing.
Normal Flow	1. Student navigates to a search result page with the desired listing. 2. Student clicks the desired listing. 3. HumbleAbode retrieves data. 4. HumbleAbode forwards the student to a listing page presenting information listed on REQ-2-1.

UC-2-2	Apply to Rent the Listing
Actor(s)	Primary: Student Secondary: Host
Description	A student applies to reserve a listing
Secondary Use Cases	UC-2-1: Access HumbleAbode Listing Page
Precondition(s)	PRE-1: Student is at a listing page PRE-2: Host sets the listing available during the desired period PRE-3: The listing is not rented yet during the desired period
Postcondition(s)	POST-1: Student applies to rent a listing POST-2: Host gets notified about the rental request
Normal Flow	1. Student navigates to a component that handles reservation requests. 2. Student selects rental period dates. 3. HumbleAbode calculates the first month’s rent and the security deposit. 4. Student requests to rent. 5. HumbleAbode forwards Student to a page for confirming the request. 6. Student can add payment information and enter a message to the host. 7. Student reviews the information they entered and presses the submit button.

3.3 Host Access

3.3.1 Description and Priority

A host must be able to list their property as a HumbleAbode listing and modify its information. This is a high-priority feature.

3.3.2 Functional Requirements

REQ-3-1: A host must be able to provide information on their property and add it as a HumbleAbode listing to be available for students to rent.

REQ-3-2: The system must collect the following property information from the host:

1. An address (Street name and number, city, province or state, country, and postal code)
2. Photos (At least one exterior and one interior photo of the property listed)
3. Contact information (At least one of: telephone number, email address)
4. Rental price of the property to be listed
5. Included utilities and amenities
6. Occupancy
7. Documentation proving that they have legal authority to rent out the property

REQ-3-3: A host must be able to additionally provide the following property information:

1. A description
2. Square footage
3. Accessibility policy
4. Pet policy
5. Smoking policy

A property will be considered to not be accessible and to not be pet-friendly unless otherwise specified.

REQ-3-4: A host must be able to select whether their property should be displayed on Airbnb, HumbleAbode, or both.

3.3.3 Use Cases

UC-3-1	Enter Property Information
Actor(s)	Primary: Host
Description	The Host provides information on their property and adds it as a HumbleAbode listing to be available for students to rent.
Secondary Use Cases	N/A

Precondition(s)	N/A
Postcondition(s)	POST-1: HumbleAbode stores information of Host’s property. POST-2: The listing is added to the Host’s listings. POST-3: HumbleAbode includes Host’s property in a search result.
Normal Flow	1. Host opens an interface for creating a new listing. 2. HumbleAbode prompts Host to enter required information. 3. HumbleAbode directs Host to an input for the property information listed in REQ 3-2. 4. Host enters the information asked on 3. 5. HumbleAbode gets confirmation from Host on required information. 6. HumbleAbode repeats flow 3. to 5. until all information listed in REQ3-2 is entered. 7. HumbleAbode allows Host to enter additional information. 8. The system directs Host an input for the property information listed in REQ 3-3. 9. Host enters the information asked on 3.1. 10. HumbleAbode gets confirmation from Host on additional information. 11. HumbleAbode repeats flow 8. to 10. until all information listed in REQ3-2 is entered. 12. HumbleAbode shows a review of all the information provided by Host. 13. Host confirms the creation of the listing. 14. The HumbleAbode listing is created with the information provided.

UC-3-2	Set Listing Preferences
Actor(s)	Primary: Host
Description	The Host modifies if the listing is only for HumbleAbode, Airbnb, or both.
Secondary Use Cases	UC-3-1: Enter Property Information
Precondition(s)	PRE-1: Host is verified on HumbleAbode. PRE-2: At least one listing is created by Host.
Postcondition(s)	POST-1: The listing is either displayed or not displayed on HumbleAbode, depending on the selected status. POST-2: The listing is either displayed or not displayed on Airbnb, depending on the selected status.
Normal Flow	1. Host opens the listing settings page. 2. HumbleAbode shows where the listing is visible and presents the options for only HumbleAbode, Airbnb, or both. 3. Host selects the desired option of where the listing is visible. 4. HumbleAbode asks Host for confirmation.

	5. Host confirms the status.

3.4 Student Accounts

3.4.1 Description and Priority

A student at a post-secondary institution must be able to create a HumbleAbode account to search for properties. This is a high-priority feature.

3.4.2 Functional Requirements

REQ-4-1: A post-secondary student must be able to create a free Airbnb account through HumbleAbode, if they do not have one.

REQ-4-2: In order to access the HumbleAbode listings, the student must provide a letter of an enrolment from their post-secondary institution. This will allow them to access all of the listings

3.4.3 Use Cases

UC-4-1	Create Account
Actor(s)	Primary: User
Description	A student without an Airbnb account creates one through the HumbleAbode UI.
Secondary Use Cases	N/A
Precondition(s)	PRE-1: User does not have an Airbnb account. PRE-2: User has an email account. PRE-3: User is attempting to use HumbleAbode services.
Postcondition(s)	POST-1: User now has an account. POST-2: User is now considered a user of Airbnb, but not verified on HumbleAbode.
Normal Flow	1. User attempts to access HumbleAbode services. 2. HumbleAbode prevents User from accessing HumbleAbode services. 3. HumbleAbode directs User to the standard Airbnb user signup process. 4. User creates an account.

UC-4-2	Verify Student
Actor(s)	Primary: User
Description	The User provides HumbleAbode with their Letter of Enrollment at a post-secondary institution. HumbleAbode in return provides access to student features on HumbleAbode.
Secondary Use Cases	UC-4-1: Create Account
Precondition(s)	N/A
Postcondition(s)	POST-1: User is now considered a Student. POST-2: User has access to HumbleAbode services.
Normal Flow	1. User attempts to access HumbleAbode services. 2. HumbleAbode prompts User to authenticate through HumbleAbode. 3. User verifies themself by uploading a Letter of Enrollment gained from a post-secondary institution. 4. HumbleAbode grants access to Student features to the user.

3.5 Host Accounts

3.5.1 Description and Priority

A user must be able to have an Airbnb account that is verified for HumbleAbode to post their property as a HumbleAbode listing for students to rent. This is a high-priority feature.

3.5.2 Functional Requirements

REQ-5-1: An existing Airbnb user, who did not provide identification to Airbnb, must be capable of being verified for HumbleAbode through providing a government-issued form of identification.

REQ-5-2: A user who has never hosted on Airbnb must be able to create an account through HumbleAbode.

REQ-5-3: A host must be able to access their account as specified in Section 3.3: Host Access.

3.5.3 Use Cases

UC-5-1	Verify Host
Actor(s)	Primary: User
Description	The User provides HumbleAbode with their government-issued form of identification. HumbleAbode in return provides access to host features on HumbleAbode.
Secondary Use Cases	UC-4-1: Create Account
Precondition(s)	PRE-1: User has an existing Airbnb account. PRE-2: User has a government-issued form of identification. PRE-3: User is not verified as a Host on HumbleAbode.
Postcondition(s)	POST-1: User has access to host features on HumbleAbode.
Normal Flow	1. User attempts to access HumbleAbode. 2. HumbleAbode asks for a government-issued form of identification. 3. User gives HumbleAbode their government-issued form of identification. 4. HumbleAbode authorizes the User on HumbleAbode. 5. User can access host features on HumbleAbode.

3.6 Student Application

3.6.1 Description and Priority

Students must fill out applications to rent on the platform. This is a high-priority feature.

3.6.2 Functional Requirements

REQ-6-1: When a student clicks on the button to apply to a listing, they must be redirected to the application page configured by the host (see section 3.10).

REQ-6-2: When a student submits all of the required questions, the application must be sent to the host.

REQ-6-3: The host must be able to approve or deny the request and contact the student to discuss further details.

3.6.3 Use Cases

UC-6-1	Apply to Listing
Actor(s)	Primary: Student Secondary: Host
Description	The Student applies to a HumbleAbode listing. Upon applying, the Student is directed to an application page configured by the Host. After filling out the application page, the application is submitted.
Secondary Use Cases	UC-10-1
Precondition(s)	PRE-1: Host has a HumbleAbode listing. PRE-2: Host has configured an application page for the listing.
Postcondition(s)	POST-1: The application is sent to Host.
Normal Flow	1. Student attempts fill out an application from the listing. 2. Student is directed to the application interface. 3. Student submits all required questions on the application interface. 4. Application is sent to Host.

UC-6-2	Review Application
Actor(s)	Primary: Host Secondary: Student
Description	The Host has a listing with a configured application page on HumbleAbode. The Host either approves or denies the submitted applications or contacts the Student about the application.
Secondary Use Cases	UC-6-1
Precondition(s)	PRE-1: Host has a HumbleAbode listing. PRE-2: Host has configured an application page for the listing. PRE-3: Student has filled out all the required questions on the application. PRE-4: The application received by Host.
Postcondition(s)	POST-1: The application is approved, denied, or waiting for the Host's decision.
Normal Flow	1. Host receives an application from a Student. 2. Host reviews all the Student's responses from the application.

	3. Host approves or denies the application, or contacts Student.

3.7 Payment

3.7.1 Description and Priority

Students must have the ability to pay their hosts through HumbleAbode. This is a high-priority feature.

3.7.2 Functional Requirements

REQ-7-1: Once a student's application has been accepted, they must pay their rent by uploading their credit card and/or Paypal information to the app. At the beginning of every month, they must be prompted to pay their rent.

REQ-7-2: Students must be able to opt-in and later opt-out to automatic payment of their rent for the rental term.

REQ-7-3: Hosts must receive the rent on the last day of each month.

3.7.3 Use Cases

UC-7-1	Process Payment
Actor(s)	Primary: Student Secondary: Host
Description	Payment is sent by the student to HumbleAbode on the first of each month, and the host receives the money on the last day of each month.
Secondary Use Cases	N/A
Precondition(s)	PRE-1: Student's application to rent with Host has been accepted by the Host.
Postcondition(s)	POST-1: Host receives the money from HumbleAbode.
Normal Flow	1. Student receives a reminder that payment is due. 2. Student pays rent. If first-time use of payment service, Student will be prompted to supply credit card or PayPal information. 3. Host receives notification at the end of the month that their payment is ready to collect. 4. Host collects their payment via chosen payout method.

UC-7-2	Opt-in / Opt-out For Automatic Payments
Actor(s)	Primary: Student
Description	A Student opts-in or opts-out at any point whether they wish their payments to be made automatically, or manually by themselves. This is reflected in whether or not their bank account is billed automatically every month or not.
Secondary Use Cases	UC-7-1: Process Payment
Precondition(s)	PRE-1: Student's application to rent with Host has been accepted by the Host. PRE-2: Student has registered their credit card or Paypal information.
Postcondition(s)	POST-1: Student automatic payment status has changed.
Normal Flow	1. Student navigates to their user profile. 2. Student enables or disables the "Automatic Payment" option. 3. Student is billed automatically every month the "Automatic Payment" option was enabled.

3.8 Reviews

3.8.1 Description and Priority

The listing page has a section for reviews from students who are either past or current renters. This is a medium-priority feature.

3.8.2 Functional Requirements

REQ-8-1: A student who rented the listing in the past must be able to post a review of the listing.

3.8.3 Use Cases

UC-8-1	Write Reviews
Actor(s)	Primary: Student Secondary: Host
Description	The student writes a review to give feedback on the listing.
Secondary Use Cases	N/A

Precondition(s)	PRE-1: Student is verified as a Student through HumbleAbode. PRE-2: Student rented the listing in the past or is currently renting it.
Postcondition(s)	POST-1: Student's post appears in the Listing Forum.
Normal Flow	1. Student views the listing page of the HumbleAbode listing. 2. HumbleAbode shows the Review Forum in the form of an embedded forum. 3. Reviews are visible to Student. 4. Student creates a new review. 5. Student posts the review. 6. Student's review is added to the Listing Reviews.

3.9 Coverage

3.9.1 Description and Priority

Hosts should have rent coverage if the student does not pay their rent on time. This is a medium-priority feature.

3.9.2 Functional Requirements

REQ-9-1: If a student doesn't pay their rent on time, HumbleAbode should notify the host and cover the rent for the month.

REQ-9-2: Students should be able to apply to have an extension for their rent payments. HumbleAbode will analyze these requests based on the student's prior renting history.

REQ-9-3: A host should be contacted regarding additional coverage on a month-by-month basis.

3.9.3 Use Cases

UC-9-1	Cover Missed Payments
Actor(s)	Primary: Host Secondary: Student
Description	If HumbleAbode does not receive the rent from the Student on time, HumbleAbode will notify the Host and compensate them in accordance with the original schedule.
Secondary Use Cases	N/A
Precondition(s)	PRE-1: There is a rental agreement in place between the Host and the Student.

	PRE-2: The Student has not applied for an extension to the payment deadline.
Postcondition(s)	POST-1: HumbleAbode pays the Host out of their insurance money. POST-2: The missed payment is visible on the Student’s internal profile with HumbleAbode.
Normal Flow	1. HumbleAbode is notified of a missed payment the next day after the due date. 2. HumbleAbode notifies the Host and Student. 3. Host is paid by HumbleAbode at the end of the month. 4. This incident is recorded on the Student’s internal profile with HumbleAbode.

UC-9-2	Apply for Extensions
Actor(s)	Primary: Student
Description	The Student applies to get an extension on the due dates for rent payments, subject to approval from HumbleAbode based on their tenancy history
Secondary Use Cases	N/A
Precondition(s)	PRE-1: There is a rental agreement in place between the student and a HumbleAbode host. PRE-2: The application is submitted at least two weeks before the payment deadline.
Postcondition(s)	POST-1: HumbleAbode makes a decision based on the Student’s history and informs them accordingly. If approved, the Student has a two weeks’ extension. If denied, Student must pay by the deadline to continue the tenancy.
Normal Flow	1. Student submits a request for extension of the payment deadline at least two weeks in advance. 2. HumbleAbode reviews the request and Student’s history. 3. Student is notified of HumbleAbode’s decision.

3.10 Application Form

3.10.1 Description and Priority

Hosts should be able to configure a customized application for prospective renters to fill out. This is a medium-priority feature.

3.10.2 Functional Requirements

REQ-10-1: As a part of setting up their listing, a host should be able to configure an additional application form for students to fill out.

REQ-10-2: The application form must support:

1. Open-ended questions
2. Checkbox elements (single- and multi-select)
3. Document uploads

REQ-10-3: The editor should be a GUI and must not require the host to know or use HTML or similar website programming languages.

3.10.3 Use cases

UC-10-1	Require an Additional Application Form
Actor(s)	Primary: Host Secondary: Student
Description	The Host asks interested applicants additional questions, through a documented process of an online form prior to finalizing the rental lease. This is optional for the Host but not for the Student if it has been enabled by the Host.
Secondary Use Cases	UC-3-1: Enter Property Information
Precondition(s)	N/A
Postcondition(s)	POST-1: The application form is visible in the listing put up by the Host.
Normal Flow	1. Host sets up their listing in accordance with UC-3-1. 2. At the end of the setup, Host has an option to opt in for an additional form with specifications outlined in REQ-10-2. 3. If opted-in, Host puts up a preferred number of questions that are either open-ended, multiple choice or require supplementary documentation upload. 4. Host previews the form and confirms the listing. 5. The additional form is published along with the listing.

4 External Interface Requirements

4.1 Interaction with the Interfaces

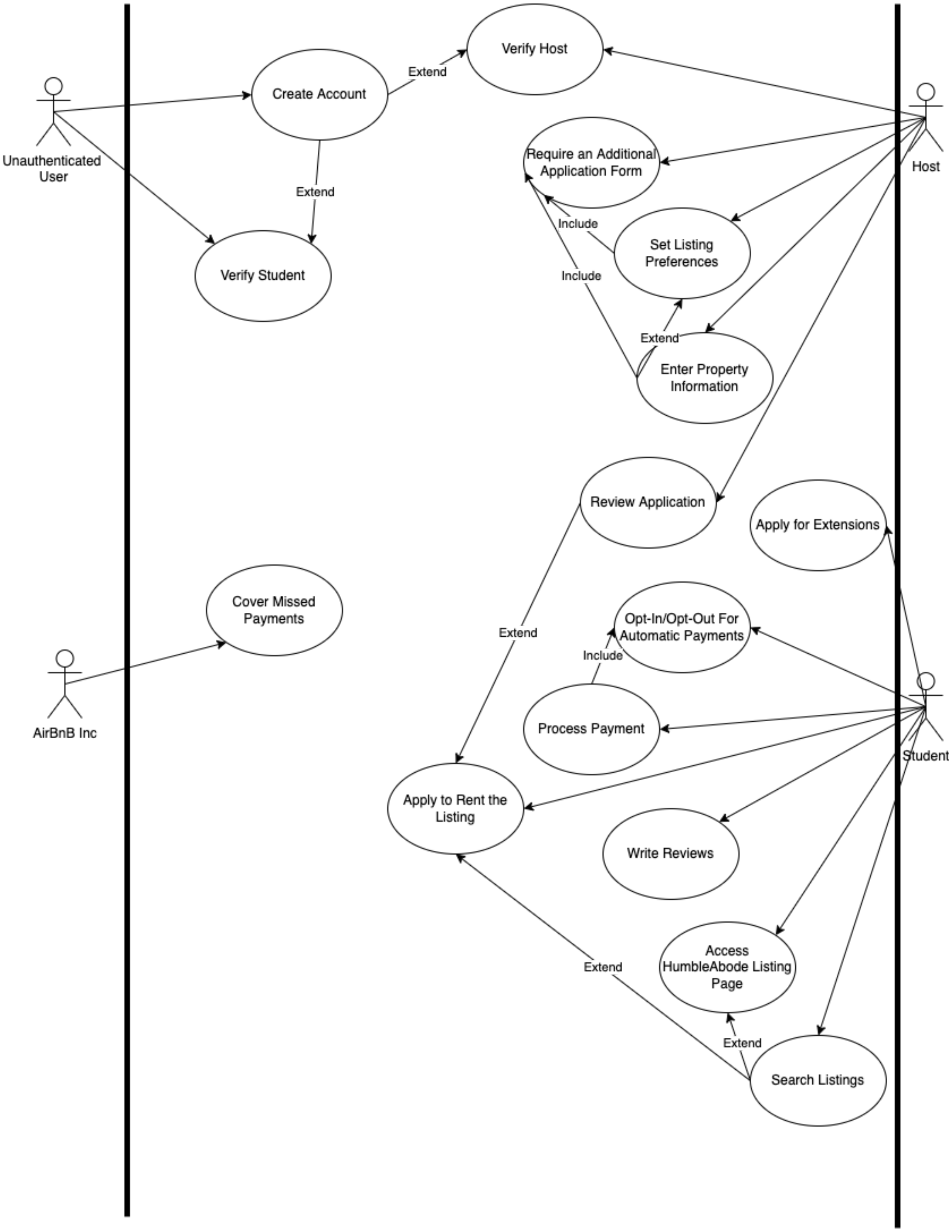


Figure 1: Use Case Diagram

4.2 User Interfaces

HumbleAdobe user interface must be consistent with the existing Airbnb user interface. To comply with Airbnb's server driven UI system, HumbleAbode shall be built on the Ghost Platform. The image below is a screen capture of the Airbnb website. Standard buttons and functions that must be implemented on HumbleAbode are shown in Figure 1 below.

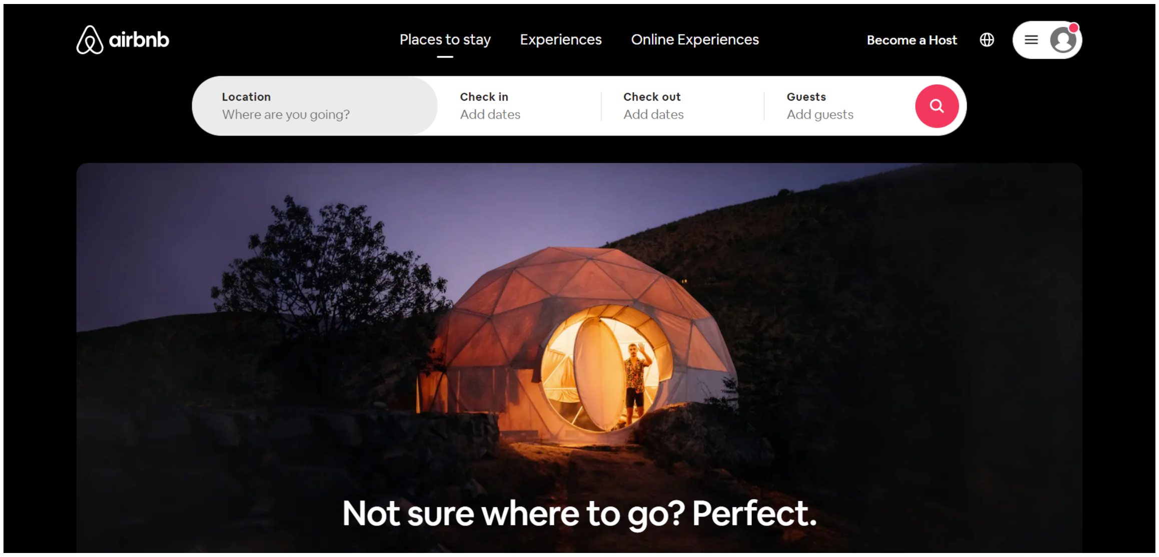
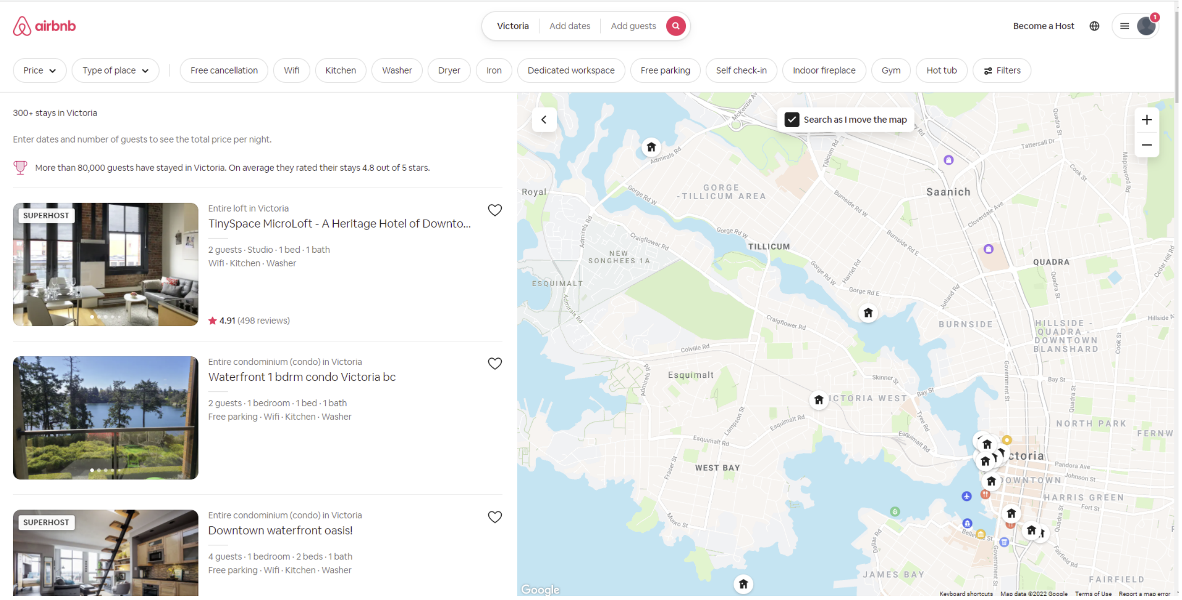


Figure 2: Airbnb's UI layout

On the right-top corner, there is a button with a user icon. When the button is clicked, it expands a dropdown with actions related to accounts, such as login, logout, notifications and messages. The search bar accepts user inputs and shows rental listings filtered by the search fields. The rental listings can be filtered further by selecting the tags below the search bar, as shown in Figure 2 below.



300+ stays in Victoria

Enter dates and number of guests to see the total price per night.

More than 80,000 guests have stayed in Victoria. On average they rated their stays 4.8 out of 5 stars.

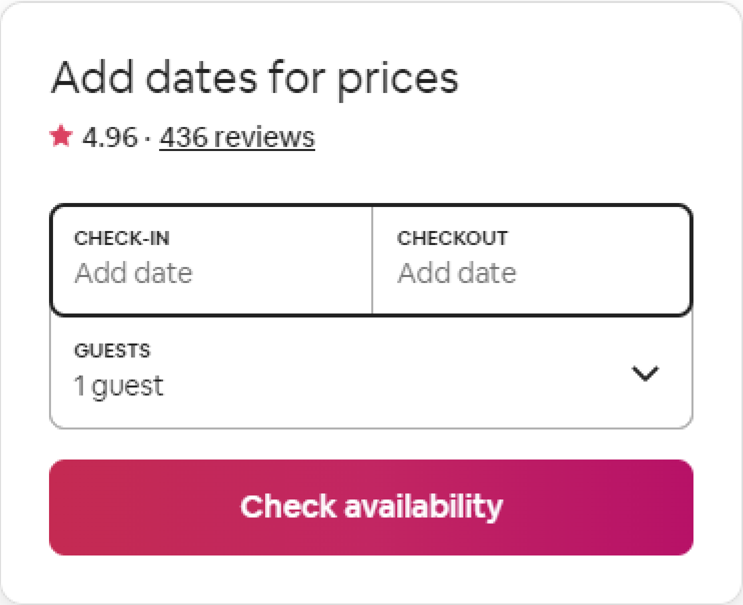
SUPERHOST
Entire loft in Victoria
TinySpace MicroLoft - A Heritage Hotel of Downto...
2 guests · Studio · 1 bed · 1 bath
Wifi · Kitchen · Washer
★ 4.91 (498 reviews)

Entire condominium (condo) in Victoria
Waterfront 1 bdrm condo Victoria bc
2 guests · 1 bedroom · 1 bed · 1 bath
Free parking · Wifi · Kitchen · Washer

SUPERHOST
Entire condominium (condo) in Victoria
Downtown waterfront oasis!
4 guests · 1 bedroom · 2 beds · 1 bath
Free parking · Wifi · Kitchen · Washer

Figure 3: Airbnb's listing search system

When the list element is clicked, a page with more information about the house is shown. There must be a component that handles reservations. With the component, a student can either check availability or select check-in/check-out dates and make a reservation. The sample image of the component is shown in Figure 3 below.



Add dates for prices

★ 4.96 · 436 reviews

CHECK-IN
Add date

CHECKOUT
Add date

GUESTS
1 guest

Figure 4: Airbnb's listing availability checker

4.3 Hardware Interfaces

The system shall be implemented on a web application and mobile application. Any hardware device that can access the supported browsers or application platforms must be able to use the system. These hardware devices include but are not limited to: laptops and desktop computers, mobile devices such as Apple and Android smartphones and tablets. HumbleAbode must be accessible via all other devices supported by Airbnb. The system must accept users' touch, click, and keyboard inputs. Any functionality (such as navigation or typing letters) must be usable with the supported hardware devices.

4.4 Software Interfaces

The supported browsers for the web application are the most up-to-date versions of Google Chrome, Mozilla Firefox, Microsoft Edge, and Opera, as they are recommended browsers for Airbnb. The mobile applications will be distributed on major application platforms, including the Apple App Store, Google PlayStore, and Samsung Galaxy Store. To be consistent with Airbnb, HumbleAbode shall use Amazon S3, EC2, and RDS for cloud storage, hosting and database. Data such as user accounts and rental listings will be shared with Airbnb's existing database.

4.5 Communications Interfaces

The HumbleAbode web application must use the HTTPS protocol for client-server communication. In addition, the system must interact with the existing Airbnb database - that is, Amazon RDS. This communication is required as the system must share its user accounts and some rental listings with Airbnb. The details of this communication, such as protocols to be used, are to be determined.

5 Other Non-Functional Requirements

5.1 Performance Requirements

Performance requirements constitute low latency and low failure rates. This is all the more imperative as HumbleAbode, like Airbnb, will follow a completely digital channel. The system must accommodate users all through the year and maintenance must be scheduled between midnight and 6am PST, during low-traffic hours. Synchronization in terms of updating listings should occur and be completed within a minute at most. The system should be able to finalize a booking within a minute from approval of an application. Synchronization in updating the availability of the listing must be instant, to ensure that only the required number of people are approved and there is no double-booking.

5.2 Safety Requirements

Hosts must have the ability to request damage and liability costs through a form with an option to add images. Students should have the ability to report a host and/or fellow renters in case of misbehavior.

5.3 Security Requirements

Privacy must be considered; HumbleAbode's listings must not be visible on Airbnb for students' safety. Regional privacy laws must be followed. Students must be able to communicate any complaints or concerns to HumbleAbode via a secure form. Student status must be verified before they can view listings and forums. Information about students already living at a place must not be visible or shared until they choose to do so themselves on forums.

5.4 Software Quality Attributes

HumbleAbode will start operating in British Columbia, Canada, but plans to eventually expand operations nationally and then globally. The system must be extensible to accommodate its regional counterparts. For example, supporting more languages and postal code formats specific to the region must be possible in a short amount of time. In addition, the system must be scalable, as it must increase resources such as instances and database size to respond to changing demands; i.e. it must not suffer performatively in the case of large numbers of new users, it must be able to handle added servers without compromising functionality. Maintainability must also be strongly considered to ensure that the new system can be easily updated and enhanced as required.

6 Other Requirements

6.1 Legal Requirements

Tenancy laws applicable to the region must be abided by. These will be included in the Terms and Conditions, and the signature will be legally binding. The Terms and Conditions must be modified accordingly when the system will be extended to a different region.

7 Data Flow Diagrams

Context Diagram (DFD Level 0)

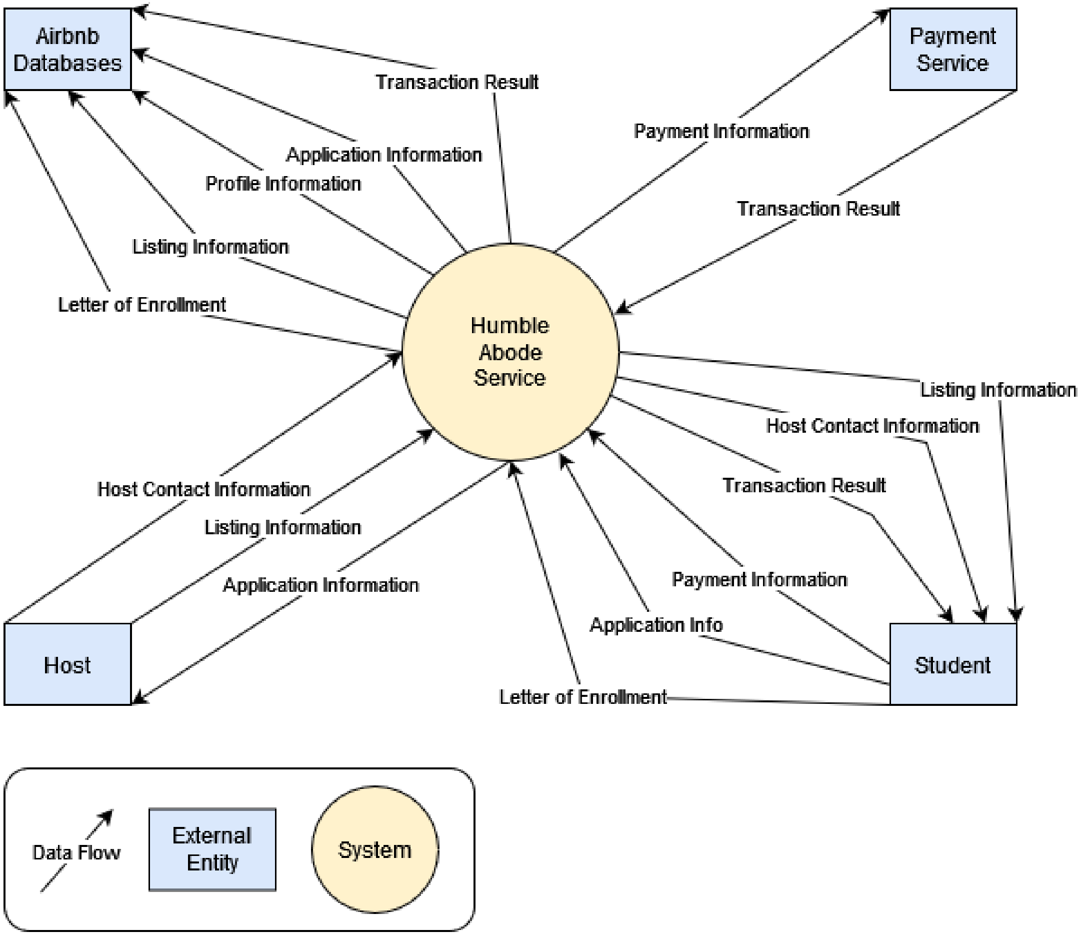


Figure 5: Level 0 Data Flow Diagram for the HumbleAbode System

DFD Level 1

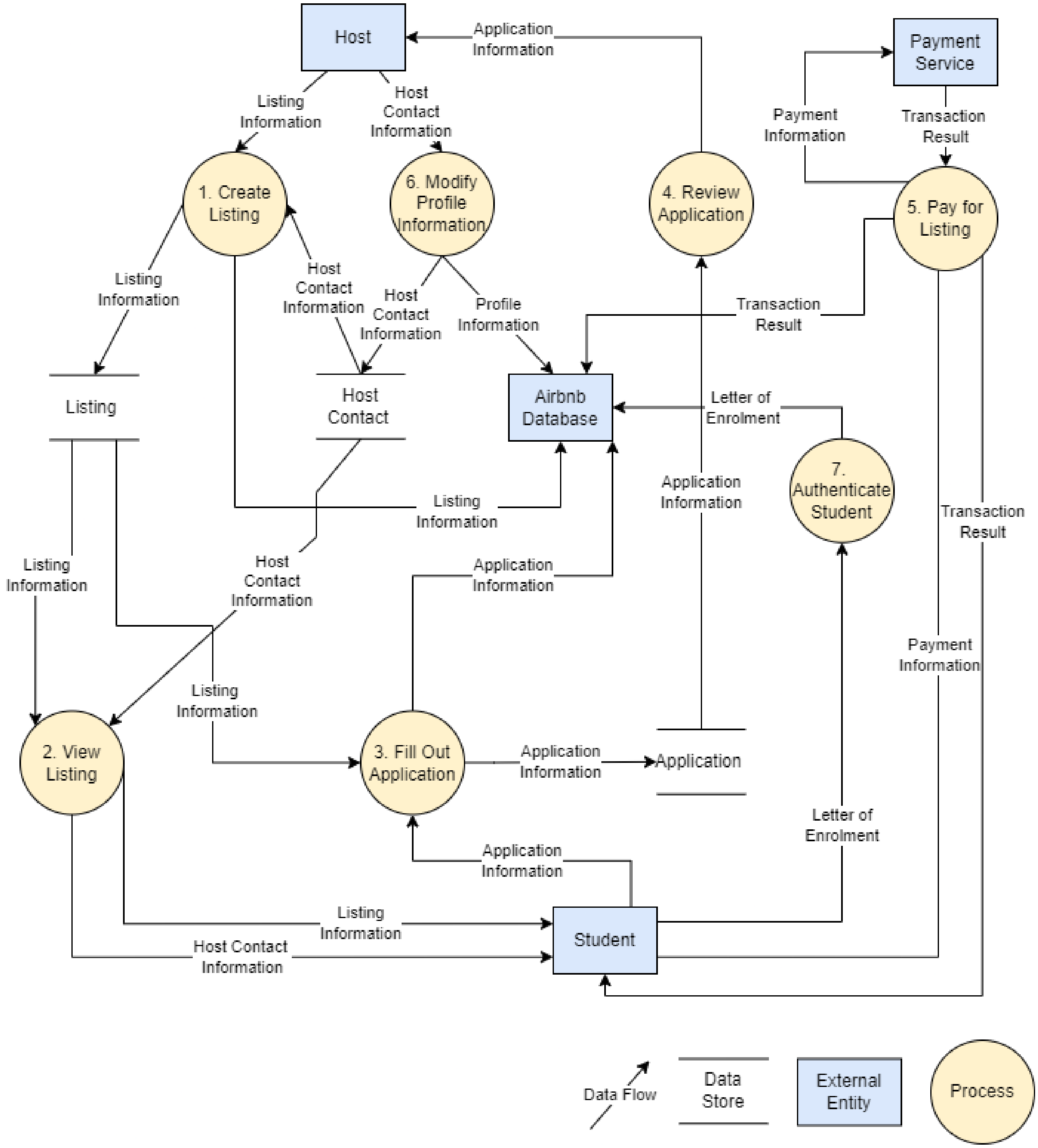


Figure 6: Level 1 Data Flow Diagram for the HumbleAbode System

DFD Level 2 (Modify Profile Information)

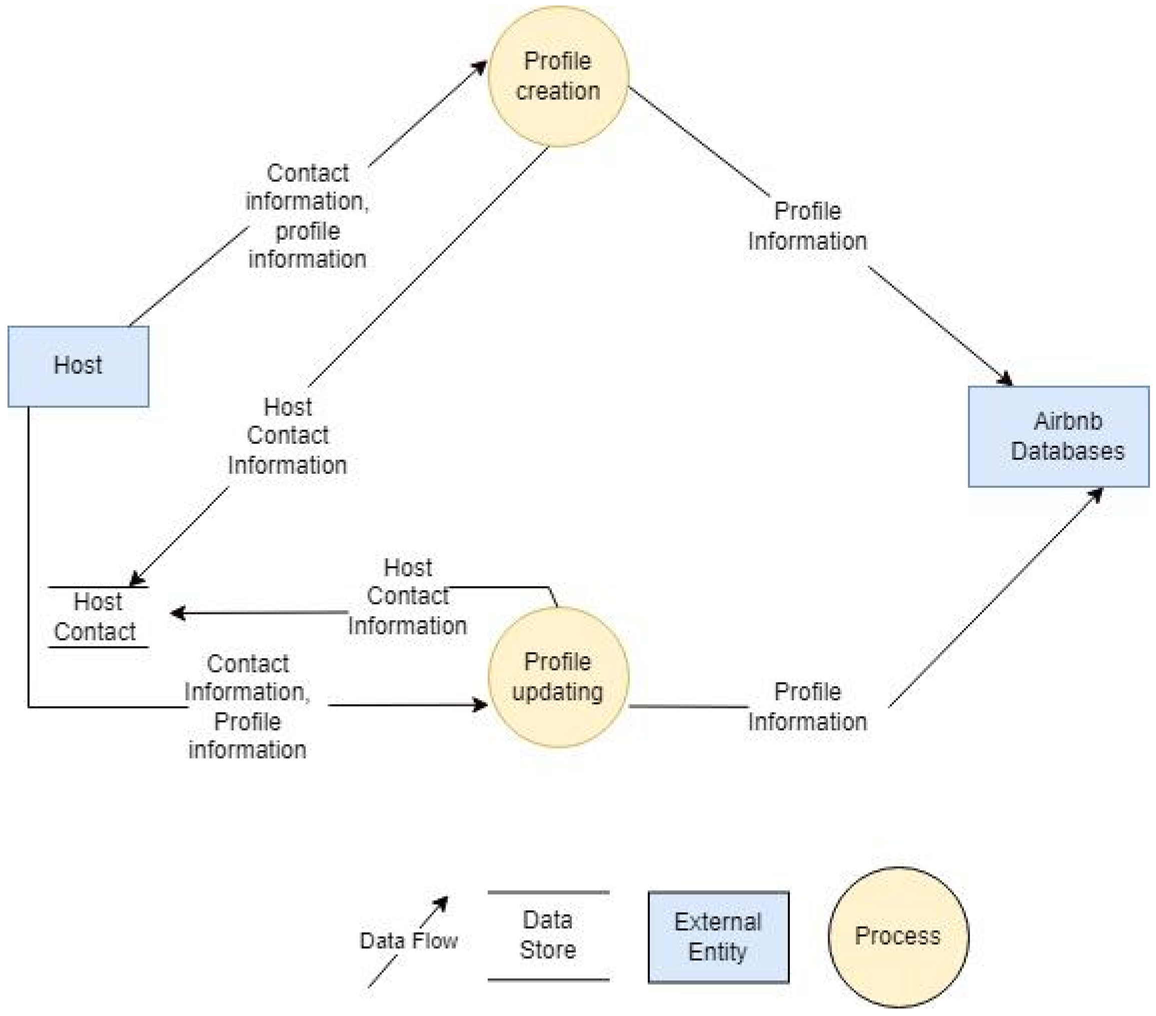


Figure 7: Level 2 Data Flow Diagram for the HumbleAbode System

8 Entity-Relationship Diagram

8.1 Entity-Relationship Diagram

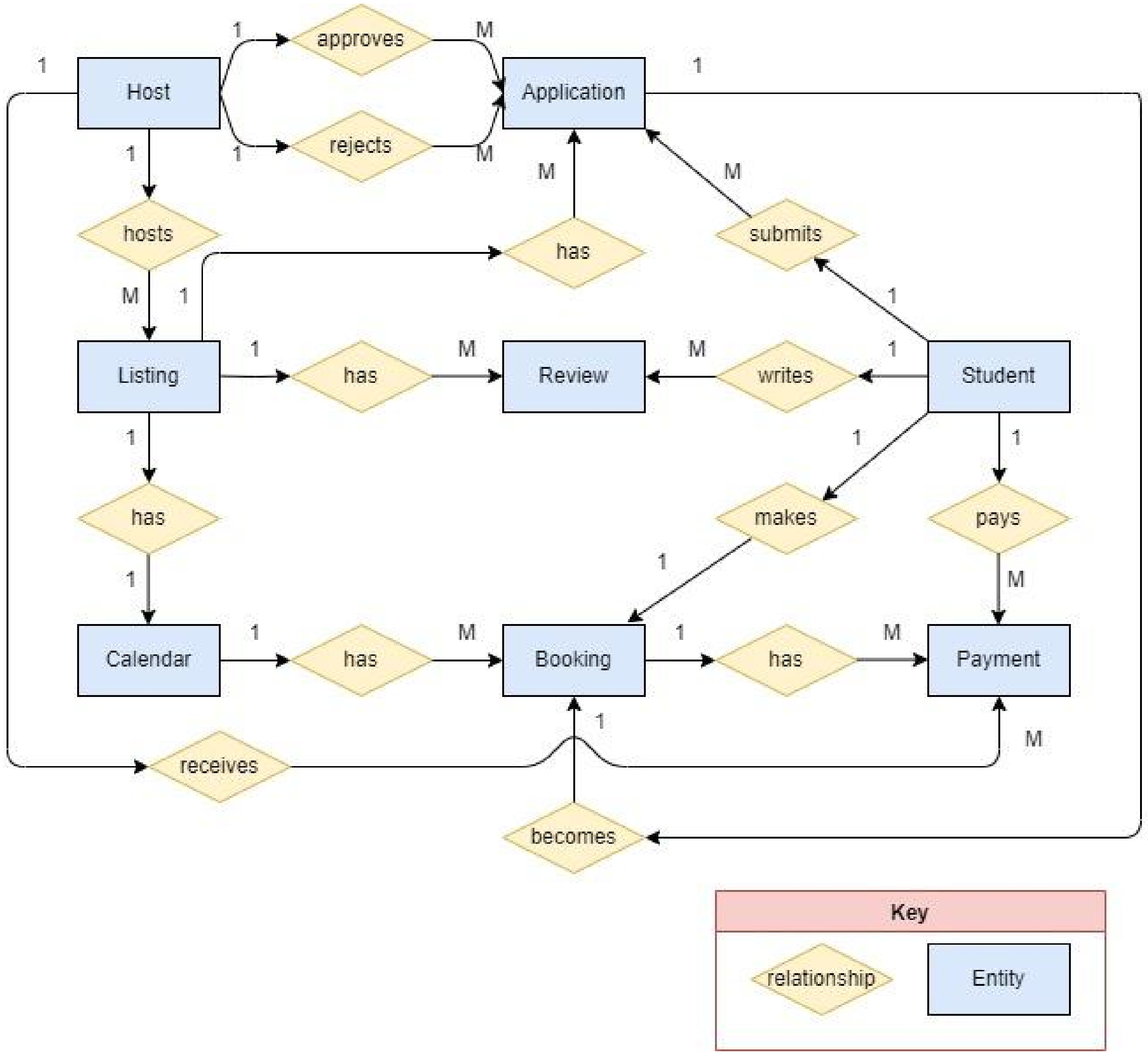


Figure 8: ER Diagram for the HumbleAbode System

8.2 Data Dictionary

Data Element	Description	Composition or Data Type	Length	Values
Host	A user who is choosing to post their property for students to rent on HumbleAbode.	*Username* Password (hashed) Government issued ID Listings Banking information		
Listing	A property available for students to rent on HumbleAbode.	*Listing ID* Address Host Calendar		
Calendar	The booking schedule for a particular listing.	*Calendar ID* Listing Bookings		
Application	A student's application to rent from a specific listing.	*Application ID* Listing Student Start date End date		
Review	A student's review of a place they rented.	*Review ID* Listing Student Content		
Student	A user who is verified as a student and is looking to rent on HumbleAbode.	*Username* Password (hashed) Proof of enrolment Contact information Applications Bookings Reviews		
Booking	A reserved	*Booking ID*		

	amount of time for a student to rent a particular property.	Listing Student Start date End date		
Payment	A student’s payment of their rent sent to the host.	*Payment ID* Listing Host Student Amount		
Username	A user’s unique identifier.	String	20 characters	Alphanumeric characters and underscores
Password	A user’s password to sign into the services.	Hashed string	20 characters	Alphanumeric characters and .,_-&$.
Government issued ID	An image of a user’s identification.	JPEG		
Banking Information	The payment information of a user.	PayPal flag PayPal username Credit card		
Listing ID	A number used to identify a listing.	Integer	11 bits	
Address	The street address of a home listed.	String	36 characters	Contains both numbers and letters
Calendar ID	A number used to identify a calendar.	Integer	11 bits	
Application ID	A number used to identify an application.	Integer	11 bits	
Review ID	A number used to identify a review.	Integer	11 bits	
Proof of Enrolment	A document uploaded by a	PDF		Contains the student’s full

	student to prove they are a university student.			name and dates of enrolment
Contact information	Information to contact a user.	*Contact ID* Phone number Text enabled Email		
Amount	The amount of money sent in a payment.	Integer	11 bits	Precisely two decimal places, in USD
Booking ID	A number used to identify a booking.	Integer	11 bits	
Payment ID	A number used to identify a payment.	Integer	11 bits	
PayPal username	A username for PayPal authentication	String	20 characters	
Credit card	Information to store a credit card	*Credit card number* Expiry date Security code		
Start date	The starting date of a duration.	Date	Standard SQL date	
End date	The ending date of a duration.	Date	Standard SQL date	
Contact ID	The number used to identify a contact.	Integer	11 bits	
Phone number	A user’s phone number.	Integer	11 digits	One digit for the long-distance code if necessary, ten digits for the phone number, including area code
Text enabled	Whether a	Boolean	1 bit	

	user’s phone number has text enabled.			
Email	A user’s email address.	String	36 characters	Contains @ and . and a valid URL for the end
Credit card number	A user’s credit card number	Integer	16 digits	
Expiry date	A user’s credit card expiry date	Date	Standard SQL date	
Security code	A user’s credit card security code	Integer	3 digits	
Content	A review of a listing.	String	1000 characters	

Appendix A: Issues List

The network protocols for communicating with the Airbnb database are yet to be determined. The user interface requirement of HumbleAbode is largely determined based on current Airbnb design. However, the design of additional features may be modified upon feedback from the client.

Appendix B: Software Tool Definitions

To avoid confusion and simplify the paragraph structure of Section 2.5, brief explanations of the various names given are explained in this appendix.

NGINX	Free, open-source web server software, used to handle many of the functions of web servers.
Rails	A framework for web application development, designed to make web application development much easier.
JavaScript	A programming language that is widely known and used, and can be applied to web page development.
React	A free, open-source JavaScript library for building user interfaces.
MySQL	Database management language for querying databases.
Sass	A preprocessor scripting language used for styling web applications.
Ghost Platform	A free, open source blogging platform which has expanded to general online publication.
Amazon RDS	A distributed relational database service by Amazon Web Services
Amazon EC2	A virtual server for running applications on Amazon Web Services.
Amazon S3	An object storage service that stores data as objects within buckets, provided by Amazon Web Services.

Appendix C: Existing Airbnb Infrastructure

Airbnb’s “Verifying your identity”

Airbnb's current policy towards requiring identification for Airbnb hosts does not necessarily always require identification as it states “we may ask for a government-issued ID or have you confirm your legal name and add your address.”