Final Documentation

University of British Columbia Okanagan
COSC 499 - Summer 2019

Prepared by

Document Owner(s)	**Role**
Stephen Okanlawon	Developer / Client Liaison / Project Manager
Jan Reisler	Developer / Integration Lead
Jae Kim	Developer / Technical Lead

Project Overview

Project Name	VRK Solutions Inc. Website design (LENTAL)		
Final Documentation Author	Jae Ung Kim & Jan Reisler & Stephen Okanlawon		
Creation Date	August 3rd, 2019	**Last Revision Date**	August 8th, 2019
Project Requestor	VRK Solutions Inc.	**Project Manager**	Stephen Okanlawon
Project Charter Status (Pending / Approve / Reject)	Pending	**Date of Project Approval**	
Proposed Project start & End Date	Start: May 2019 End: August 2019		

Approved By

_____ _____

Scott Fazackerley Date

_____ _____

Client Date

Scope and Charter

Project

Purpose and Objectives

The purpose is to build a website that serves as a portal where landlords can advertise their available home rental units for tenants to view and inquire about from landlords. Users can login to rent out a suite/house or seek residence. The website will be simple and intuitive as to avoid clutter and to make the site more appealing to users so they do not feel that the site is too complicated and overwhelming. It will allow users to filter out listings to find specific listings that they are interested in, as well as providing contact information and a wide range of information for each listing. Navigation between pages will also be simple and easy.

Related Success Criteria

The project will be successful if the website meets all the requirements and features that the client has listed. The website should allow users to search through the listings efficiently using features like; map searching via distance, scroll bar searching for different price ranges, and searching through other filters like smoke-friendly, pet-friendly, amongst others. We also want the website to be agile so that the website updates in real-time and can also be modified over time.

Project Team

- Jae Ung Kim
- Jan Reisler
- Stephen Okanlawon

Stakeholders

- Team
- Community members
- Customers
- Valentin Koch/ Sven Stein
- UBC

Requirements

Functional Requirements

- System will allow users to sign up and store the sign up information
- System will allow users to log in
- System will allow users to sign up as tenant or landlord and switch between them effortlessly
- System will allow users to view listings on map based on location and distance
- System will allow users to enter property information and post ad
- System will allow users to filter properties to find specific listings
- System will offer admin login via an admin portal
- Admin portal gives admin ability to delete user account and/or ads
- UI is clear and only includes relevant info on each page and groups all relevant data together
- System will offer password recovery methods
- System allows general search of listings without login

Non-Functional Requirements

- Listings must load and be displayed in less than two seconds when a user searches for specific listings or is filtering them
- Website has responsive design
- Users only need email, name, and password to register an account
- Logging into the website must take less than one second
- When creating listings or an account, all fields must be correctly handled of any range of characters and empty fields

Technical Requirements

- The website will be running on an online server
- The website front end will be created with CSS, HTML, and JavaScript/JQuery
- The website backend will be created with PHP and AJAX connected to a MySQL database
- Website can be accessed and correctly viewed with any mobile and desktop updated browser
- MVC design pattern

User Requirements

- User will be able to search all listings without login.
- User will be able to log in

- User will be able to sign up
- User will be able to sign up as tenant or landlord and switch between them
- User will be able to view listings on map based on location and distance
- User will be able to enter property information and post ad as a landlord
- User will be able to filter ads to find specific ads as a tenant
- Admin will be able to delete user account and/or ads via the admin portal
- User will be able to recover password

Admin Requirements

- Allow admins to add users upon request
- Allow admins to delete users upon request
- Allow admins to update users upon request
- Allow admins to delete listings upon request
- Allow admins to recover user account
- Allow admins to recover user listings
- Allow admins to recover user email and passwords
- Allow admins to view user and listing

Assumptions and Constraints

Assumptions

- Developers are comfortable programming in PHP.
- Developers are comfortable using GitHub.
- Developers will be provided with necessary materials from the client to complete the project.
- Developers will complete the project in a timely manner during the duration of the project (13 weeks).
- Client will respond to the developers requests or concerns as soon as possible.
- Developers will not run into technical problems.

Constraints

- The developers and clients will be away on vacation during the summer.
- Tight time constraints - the developers only have 13 weeks to complete the project.
- Knowledge constraints - the developers are 4th year University students.
- Some of the features clients require might be hard to implement for the developers.
- User testing difficulties - After the deployment of the website, getting users to participate in user testing can be difficult.

Project Description and boundaries

Description

- Once the project is completed, we will have a complete free and working website similar to trivago and airBnB but for rental properties. Our website will allow landlords to first rent out their properties but they can later decide to list the properties for sale if they want to. There will be lots of features that are helpful for all users of our website. For example, similar to trivago we will allow users to search listings via map so users can see the exact locations of listed properties. There will be more features implemented to this map search to guide users to find the best possible find for them.

Boundaries

- The project has to be finished prior to August 9th, 2019. However, the code has to be finished by August 2nd, 2019 so developers can run tests and fix bugs if any.

High Level Risks

- Cost risks: The project is done by University students (developers) as their graduation requirements; thus there is absolutely no cost. However, since it is also for actual company's website, developers might have problems with features that need subscriptions.
- Schedule risks: developers or client can get sick and delay the project process. Developers might need more time than expected time for certain features.
- Performance risks: Some of the website features might be too hard to implement for University students.
- Lack of communication between developers and the client.
- Poor team organization and team performance.
- Conflicts might arise between team members.
- Missing people

Budget Requirements

- Monetary value for this project is $0 since all necessary materials will be provided from the client.
- Developers will each spend at least 16 hours per week for 13 weeks on the project.
- IDE we have chosen (atom) is free; thus no cost to implement it.

Environmental Constraints

- Extreme heat environment in Kelowna might cause developers to get ill.
- University WIFI might have connectivity issues resulting in delay in development.
- Website server might go down.
- Power outage in the middle of developing process might affect unsaved data. (If any)
- Website could be hacked or intercepted.

Development Process

Communication plan	When	What	Who
	May 22 2019	First meeting with the client	Jae & Stephen & Jan
	May 31 2019	Scope & Charter Requirements	Jae & Stephen & Jan
	June 7 2019	Design documents	Jae & Stephen & Jan
	August 8 2019	Final presentation	Jae & Stephen & Jan

Development Standards and tools

- During weekly meetings, the project team's integration lead, Jan, will make sure each team member made some progress with coding.
- During weekly meetings, integration lead will run a linter program to ensure that codes from the team follow the same style guidelines.
- Once everything checks out and merged to master branch, the project team will go through multiple testing (testing if certain features work with another) to ensure our latest version is working.
- As for version control, GitHub will be used. Every week, integration will make sure all of our codes are committed and pushed to the master branch at least once a week, and the master branch will always maintain the latest working version of the code.
- Different branches will represent different features of the website.

- Trello will be used to keep track of who is working on which features.
- Every merging to master branch will be documented by the technical lead.
- There will be a weekly meeting to ensure the project is making progress every week.

Work Breakdown Structure

Task List	**Estimated Hours**		
	Jae	**Jan**	**Stephen**
1.0 Documentation			
1.1) Scope and Charter	15	5	5
1.2) Weekly Reports	13	10	13
Total number of Hours Assigned Per Team Member	23	15	18
Weekly Average of Hours Assigned Per Team Member (13 weeks)	1.8	1.15	1.4
2.0 Learning			
2.1) Developers should review HTML / CSS	2	1	3
2.2) Developers should review PHP	2	2	6
2.3) Developers should review GitHub	1	0	3
2.3) Developers should learn how Bootstrap works	4	3	6
2.4) Developers should review Basic networking (server, XAMPP, etc)	2	1	5
2.5) Weekly Team meetings (Cumulative)	13	13	13
Total number of Hours Assigned Per Team Member	24	20	36
Weekly Average of Hours Assigned Per Team Member (13 weeks)	1.9	1.5	2.8

3.0 Designing			
3.1) Designing the layout of front page	2	2	2
3.2) Designing the layout of account pages	2	2	2
3.3) Designing name & logo of the company	5	1	
3.4) Designing the layout of map search page	2	2	5
3.5) Data models and diagrams (UML, ER, etc)	5	5	7
3.6) Designing the test plan	2	4	1
3.7) Finding correct/best tools to use for project	1	1	1
Total number of Hours Assigned Per Team Member	19	17	18
Weekly Average of Hours Assigned Per Team Member (13 weeks)	1.5	1.3	1.4
4.0 Developing the website			
4.1) Building front-end UI pages with bootstrap and make mobile compatible (Home page, browse page, admin page, account page, create listing page)	15	20	10
4.2) Building Database and make accessible to website	10	10	20
4.3) Integrate and display map engine on browse page	20	20	15
4.4) Implement functioning accounts and be able to login, register, post listings, and contact landlord correctly	20	25	20
4.5) Implement search function which displays listings and can be narrowed down with filters	15	25	15
Total number of Hours Assigned Per Team Member	80	100	80
Weekly Average of Hours Assigned Per Team Member (13 weeks)	6.1	7.7	6.1
4.0 Testing			

4.1) Integration Testing	5	10	5
4.2) User Testing	5	7	6
4.3) Usability Testing	5	7	5
4.4) Test plan for website	5	5	5
4.4) General	5	5	5
Total number of Hours Assigned Per Team Member	25	34	26
Weekly Average of Hours Assigned Per Team Member (13 weeks)	1.9	2.6	2.0
5.0 Other (Technical problems, missed details, change of requirements, variability, etc.)	30	30	30
Total number of Hours Assigned Per Team Member	30	30	30
Weekly Average of Hours Assigned Per Team Member (13 weeks)	2.3	2.3	2.3

Design and Testing

Project Description

Our website, Lental, will allow users to post ads/listings for their suites and homes and search for ads/listings to rent. The idea is to have a simple design that is very user friendly and easy to find homes/suites to rent. It will have many filtering options to find a specific listing, a map to show listing locations and distance, allow for posting ads/listings, logging into an account and save listings to view later, and even a search agent that automatically emails listings to users with specified filters.

Technical Specifications

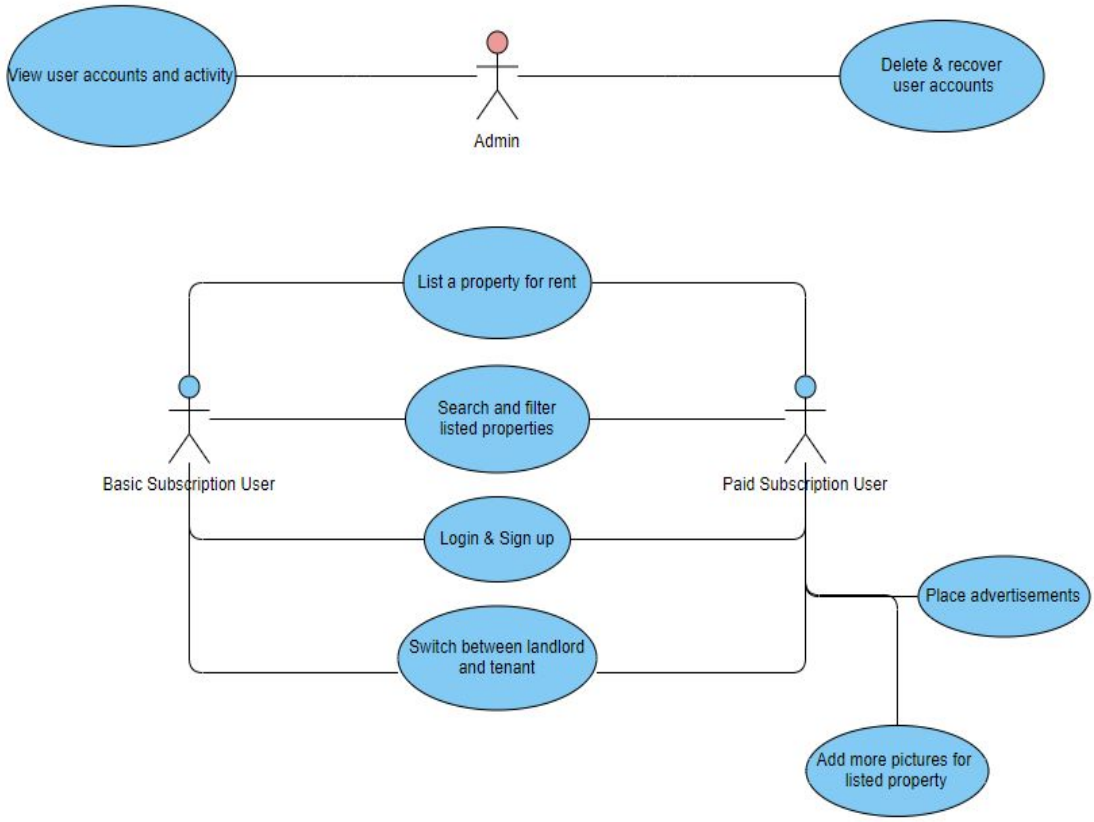
- HTML, CSS, and JavaScript/JQuery will be used for the front end and bootstrap will be used as the framework
- PHP will be used as the backend with no framework
- PHPUnit will be used to test the PHP and Selenium will be used to test the website in a more automated way
- Atom will be used as the main IDE, GitHub for version control, and Laragon for local server/database hosting for offline development
- MySQL will be used for the database
- Google maps or Bing maps will be used for maps view

User Groups & Usage Scenarios

There will be three main user groups for this website namely:

- Admin: The admin will be granted access to the back end of the database so they will be able to view, update, delete and recover accounts and account information from the website so that the website can be constantly monitored in case of spam, inappropriate posts or errors by the customer.
- Basic (free) user: The basic/free user will be granted access to listing properties and viewing properties but some of the features will be limited
- Paid user: The paid user is similar to the basic user except, the limitations are lifted and give the user more access to the website's features.

Below is a visual representation of some usage scenarios for the aforementioned user groups:



Use case index:

Use Case ID	Use Case Name	Primary Actor(s)	Complexity (1-5)	Priority (1-5)
1	View user accounts and activity	Admin	2	4
2	Delete & recover user accounts	Admin	4	5
3	List property	Basic and Paid	3	5

	for rent	User		
4	Search and filter listed properties	Basic and Paid User	3	4
5	Login	Basic and Paid user	2	5
6	Sign up	Basic and PAid user	2	5
7	Switch between landlord and tenant	Basic and paid user	4	2
8	Place advertisements	Paid users	2	2
9	Add more pictures for listed property	Paid users	4	1

Use Cases:

ID:	1
Name:	View accounts and activity
Secondary actor(s)	Admin portal
Flow of Events	1. Sign in to admin portal 2. Select view accounts 3. Accounts are displayed by system
Pre-conditions	1. Access to admin portal 2. Signed in as administrator
Post-conditions	1. Access to view all accounts in database 2. Access to view listings and ads on website

Descriptions	Admin views all accounts and account activities/listings and ads

ID:	2
Name:	Delete and recover user accounts
Secondary actor(s)	Admin portal
Flow of Events	1. Search for user 2. Select user account 3. Delete user account
Pre-conditions	1. Access to admin portal 2. Signed in as administrator 3. Able to view accounts and listings
Post-conditions	1. Can delete accounts and listings 2. Can recover accounts and listings
Descriptions	Admin can delete and recover user accounts as well as listings

ID:	3
Name:	List property for rent
Secondary actor(s)	
Flow of Events	1. Click add listing button 2. Fill out form with required listing information 3. Click submit button 4. Receive confirmation of successful listing
Pre-conditions	1. Access to website 2. Have an account 3. Logged in
Post-conditions	Listing added to the website and viewable
Descriptions	User is able to add a listing to the website provided they have a

	valid account

ID:	4
Name:	Search and filter properties
Secondary actor(s)	
Flow of Events	1. Click on search bar 2. Type location/city into search bar 3. Properties in city are displayed 4. Select filters to narrow down preferences 5. Click on search 6. New properties with filtered attributes displayed
Pre-conditions	1. Internet access 2. Website page
Post-conditions	Properties displayed with filtered attributes
Descriptions	User is able to filter down properties in certain locations

ID:	5
Name:	Log in
Secondary actor(s)	
Flow of Events	1. Click on email box 2. Fill out email 3. Click on password box 4. Fill out password 5. Click login 6. User logged in
Pre-conditions	1. Internet access 2. Website page
Post-conditions	User is then logged in
Descriptions	User is able to log in with account and utilize more of the perks

	available

ID:	6
Name:	Sign up
Secondary actor(s)	
Flow of Events	1. Click on first name 2. Fill out form 3. Click on last name 4. Fill out form 5. Click on email 6. Fill out form 7. CLick on password 8. Fill out form 9. Click on confirm password 10. Fill out form 11. Click register
Pre-conditions	1. Internet access 2. Website access
Post-conditions	User is then registered and can log into website
Descriptions	User is able to register an account

ID:	7
Name:	Switch between landlord and tenant
Secondary actor(s)	
Flow of Events	1. No action needed, account can perform both landlord and tenant functions at same time
Pre-conditions	1. Has an account 2. Logged in
Post-conditions	

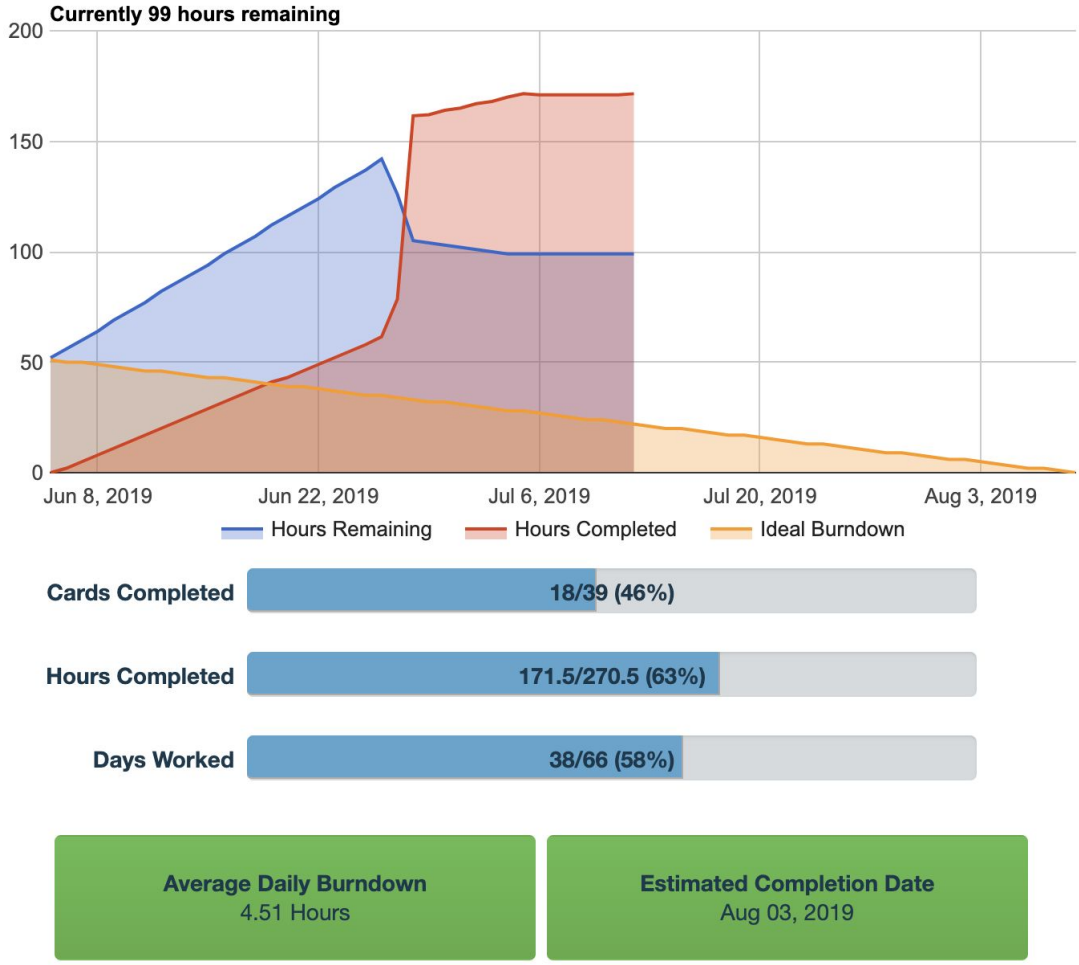
Descriptions	User is then able to access tenant or landlord features

ID:	8
Name:	Place advertisements
Secondary actor(s)	
Flow of Events	1. Click add listing 2. Fill out form 3. Submit
Pre-conditions	1. Has an account 2. Is logged in
Post-conditions	1. Listing is added to site 2. Able to view listing from search page
Descriptions	User is able to add a listing to the site and view it.

ID:	9
Name:	Add more pictures for listed property
Secondary actor(s)	
Flow of Events	1. Click on listing 2. Edit listing 3. Add pictures 4. Submit
Pre-conditions	1. Has paid account 2. Is logged in 3. Listing exists
Post-conditions	New pictures added to listing
Descriptions	Users with paid subscriptions can add more pictures to a pre-exisiting listing

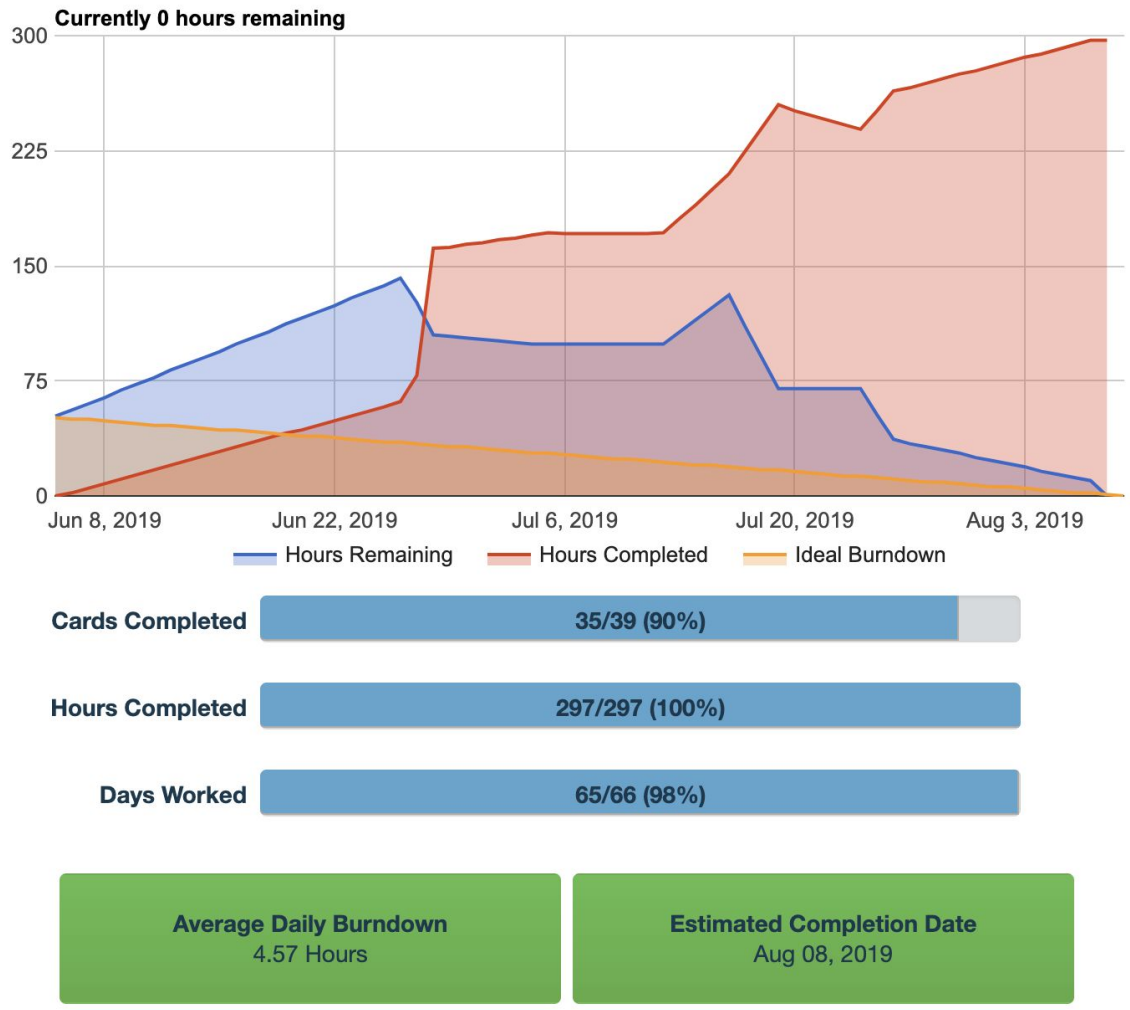
Burnup / Burndown

Burndown - Friday, July 12nd, 2019



A burndown in the middle of project is shown above. By implementing feedback from user testing and clients, adjustments were made to make sure the project could be done on time. Estimated Completion date for the project was August 3rd, which was 6 days prior to the due date.

Burndown - Thursday, August 8th, 2019



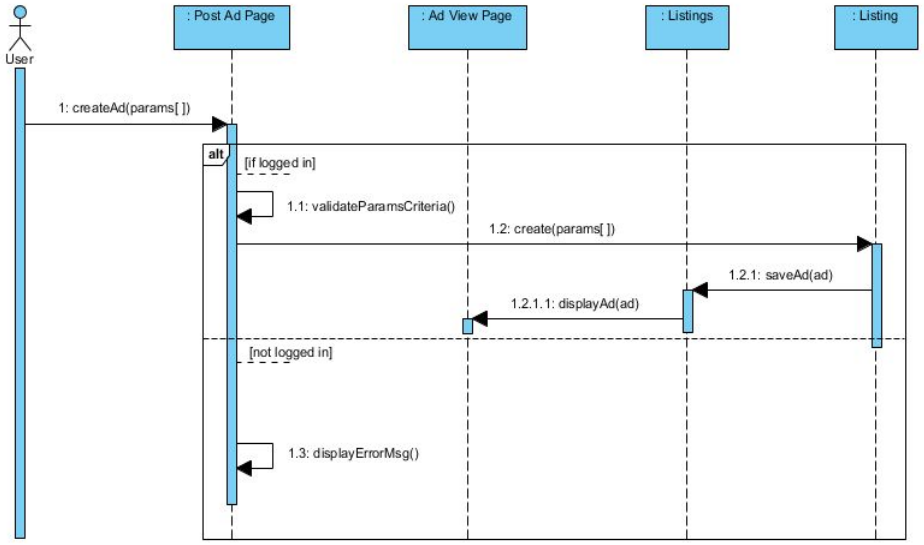
A burndown nearing the end of the project is shown above. All of requirements were done. Our estimated completion date was August 3rd, but due to circumstances, it had to be postponed to August 8th, which was a day prior to the due date.

System Architecture Diagrams and Models

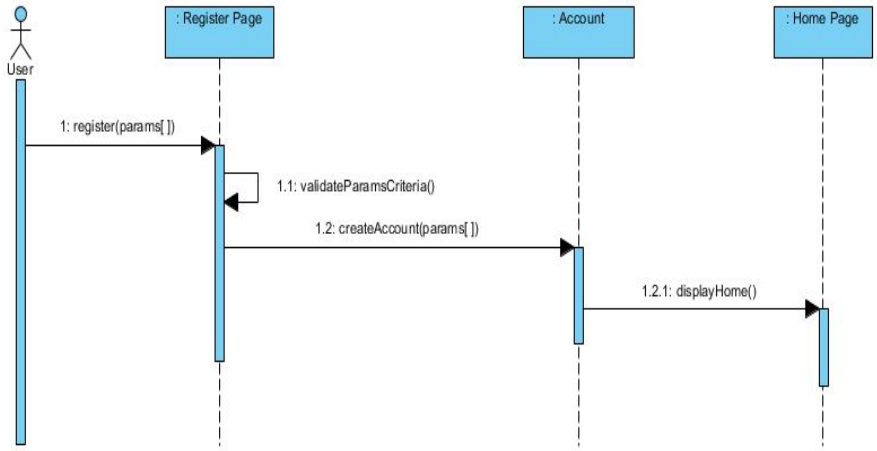
Sequence Diagrams

UML Sequence Diagrams to detail how actions are carried out based on user interaction

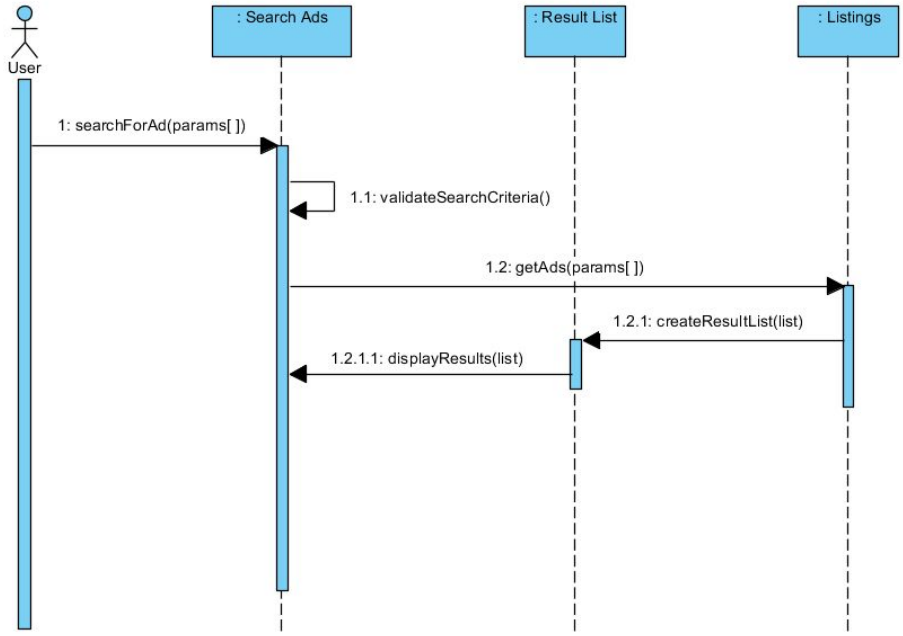
- The Post Ad sequence diagram shows the operations involved when posting an ad/listing. This sequence diagram also follows a very similar structure to editing an ad/listing.



- The Register sequence diagram shows the operations involved in registering for an account.

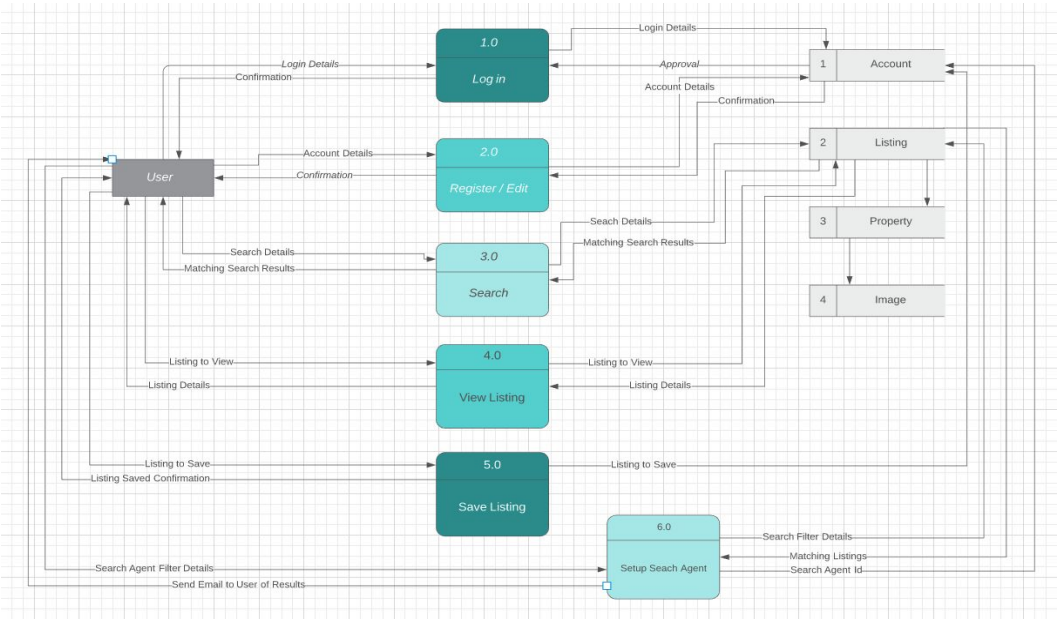


- The Search For Ads sequence diagram shows the operations involved in searching for and filtering ads/listings.



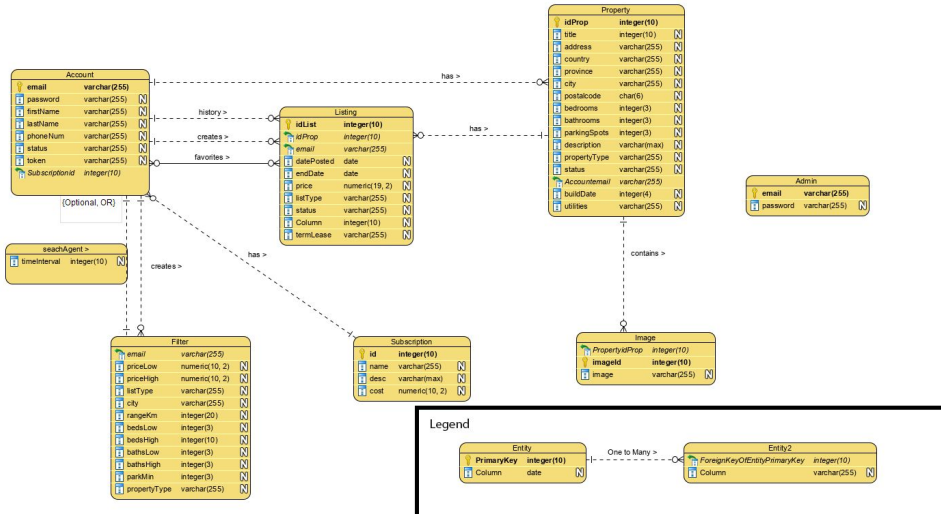
Data-Flow Diagram

Data-Flow Diagram to show how data is processed and created in the system



ER Diagram

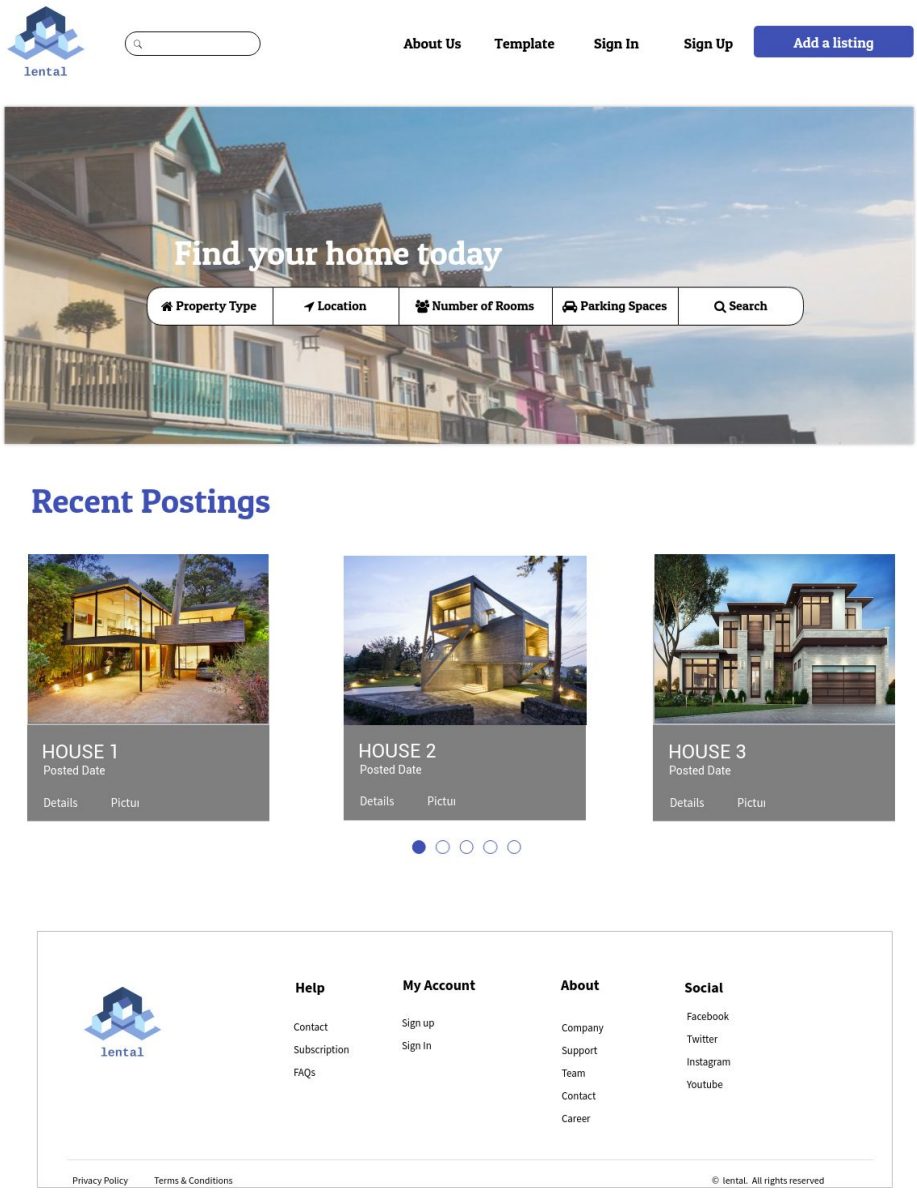
ER Diagram (Entity Relationship Diagram) to show how entities relate to each other in the database



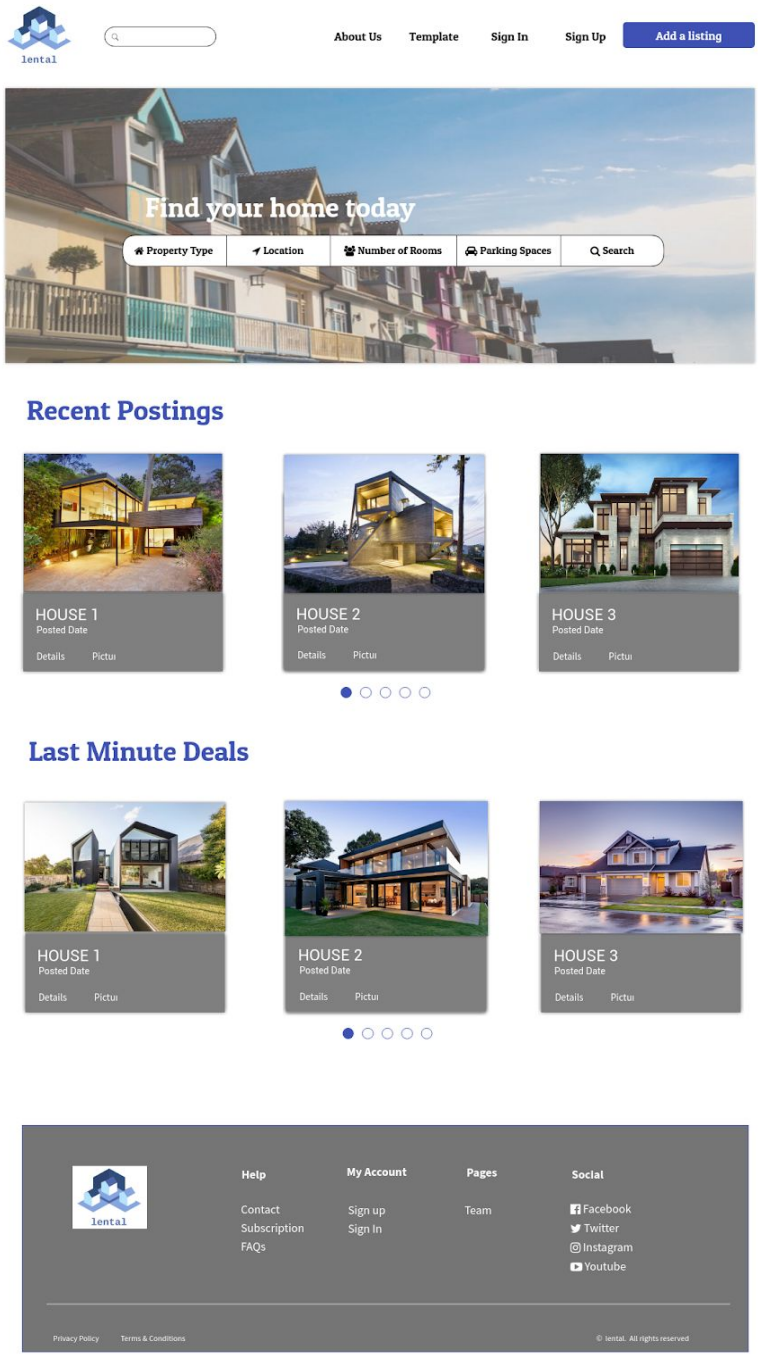
UI Mockups

Front Page

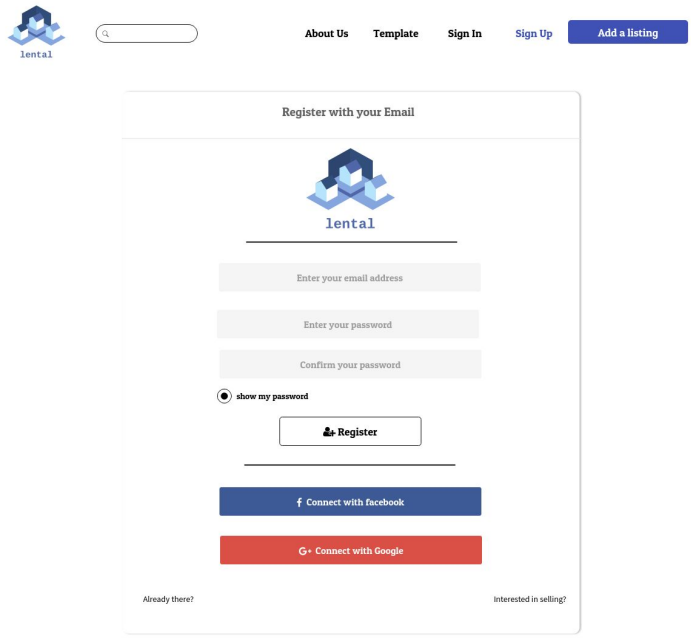
Initial



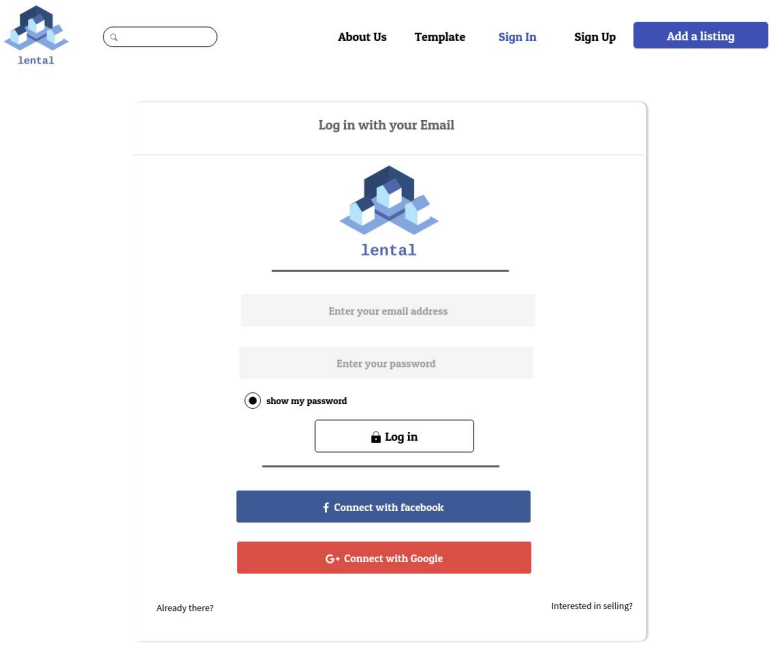
Adjusted


lental
About Us
Template
Sign In
Sign Up
Add a listing
Find your home today
Property Type
Location
Number of Rooms
Parking Spaces
Search
Recent Postings
HOUSE 1
Posted Date
Details
Pictur
HOUSE 2
Posted Date
Details
Pictur
HOUSE 3
Posted Date
Details
Pictur
Last Minute Deals
HOUSE 1
Posted Date
Details
Pictur
HOUSE 2
Posted Date
Details
Pictur
HOUSE 3
Posted Date
Details
Pictur
lental
Help
Contact
Subscription
FAQs
My Account
Sign up
Sign In
Pages
Team
Social
Facebook
Twitter
Instagram
Youtube
Privacy Policy
Terms & Conditions
© lental. All rights reserved

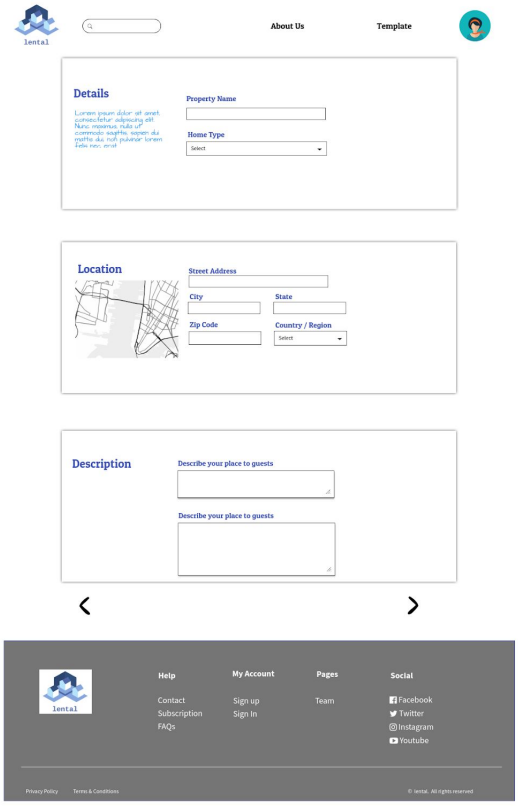
Sign up

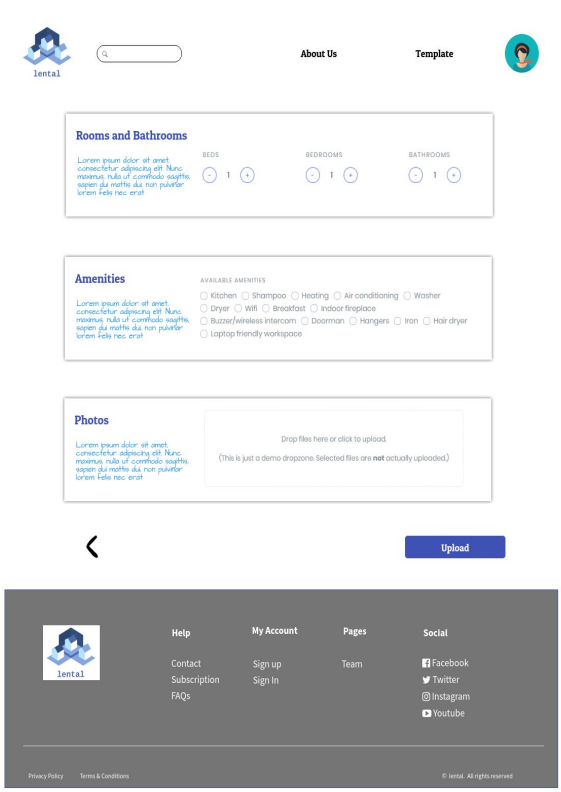


Register



Adding Listings





Testing

Functionality Testing

Check all the links in web pages

- Test the outgoing links from all pages to the specific domain under test.
- Test links jumping on the same pages
- Test to check if there are any orphan pages
- Check for broken links

Check forms on all pages

- Check validations on each fields.
- Check how forms respond to SQL injection, wrong inputs, and etc.

Component Testing

Validate HTML, CSS, and JavaScript

- In order to test for syntax errors for HTML & CSS, we will run our markups through W3C validator.
- Unit testing will be conducted for each JavaScript functions.

Browser Compatibility

- We will check if our website work with different types of browsers.

Integration Testing

Check how all different features & pages work together

- Before merging into master branch, we will run our codes through the CI tool (TravisCI) to test whether our components behave as expected.
- We will combine the individual tested components one by one and test incrementally.

User Testing

User walk-through of website (mobile & browser)

- We will have a random person go through the website attempting to complete a list of tasks such as “create an account and sign in.”
- Once the user does that, we can ask him / her to “post a listing in Kelowna.”

Test Report

	Requirements	Type of Testing (I: Integration Testing, S: System Functionality Testing, O: Operational Acceptance Testing, UN: Unit Testing, US: Usability Testing, A: Acceptance Testing)	Pass or Fail (P: Pass, F: Fail)	Contributor: (JK: Jae Ung Kim, JR: Jan Reisler, SO: Stephen Okanlawon)
FUNCTIONAL REQUIREMENTS				
1 - Home				
1.1	Users will be able to search for properties by city	UN	P	JR
1.2	Users will be able to view their previously searched listings			
2 - Find Listing				
2.1	Users will be able to use sliders, buttons, and input fields to filter listings to their specification	A	P	JR,SO,JK
2.2	Users will be able to navigate through listings and select a listing for a more detailed view	A	P	JR,SO,JK
2.3	The website will search for and update the listings based on user input	A	P	JR,SO,JK
2.4	The website correctly retrieves listings from the database	UN	P	JR
3 - Register				
3.1	The user will be able to register an account on the website	A	P	JR,SO,JK
3.2	The website will store user account on the database	UN, I	P	JR
3.2	The website will validate user data	A	P	JR,SO,JK
4 - Login/Account				
4.1	The user will be able to login to their account	A	P	JR,SO,JK
4.2	The website will validate user login and allow the user to log in	UN, I	P	JR
4.3	The user will be able to recover their account password using their email	A	P	JR,SO,JK
4.4	The user can logout and the session will be destroyed	I	P	JR
4.5	The website will check if a user is logged in and redirect if not logged in on certain pages	I	P	JR

5 - Add Listing				
5.1	The user will be able to add a listing and upload images for the property	A	P	JR,SO,JK
5.2	The website will store a users listing to the database and connect it to the users account	UN	P	JR
5.3	The user will only be able to add a listing if they are logged in	A	P	JR,SO,JK
5.4	The website will validate the users inputs	A	P	JR,SO,JK
6 - View Listing				
6.1	Users will be able to view multiple images for specific listing and listing details	A	P	JR,SO,JK
6.2	The website correctly retrieves listing data from the database	UN	P	JR
7 - Navigation and Links				
7.1	Users will be directed to the register page through navbar	A	P	JR,SO,JK
7.2	Users will be able to log in through navbar with ajax and not have to leave the page they were on	UN	P	JR
7.3	Users will be directed back to front page when they click on the logo	A	P	JR,SO,JK
7.4	The website will update the navbar if the user is logged in to show the appropriate links	A	P	JR,SO,JK
8 - Admin				
10.1	Admins will be able to view all accounts, listings, filters, and subscriptions	A	P	JR,SO,JK
10.2	Admins will be able to change/delete accounts, listings, filters, and subscriptions	UN,A	P	JR,SO,JK
10.3	Admins will have a separate login table stored on the database and a seperate website for viewing	UN, A	P	JR,SO,JK
10.4	The website will change/delete accounts, listings, filters, and subscriptions on the database by admin request	UN	P	JR
9 - Database/Server				
11.1	The system will request for and store data in the database via web page requests	UN	P	JR
11.2	The system will store all user accounts	UN	P	JR
11.3	The system will store all properties/listings posted by users	UN	P	JR
11.4	The system will store all images for properties/listings	UN	P	JR
11.5	The system will use an api (databaseAPI) to connect to the database and fulfil server requests	UN	P	JR
10 - Maps				
12.1	Users will be able to view and navigate a map of listings			
12.2	The map will update its listings based on user filter options and the maps focused area			

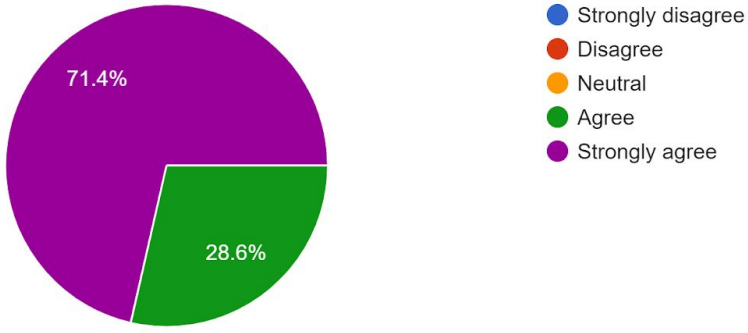
11 - About				
13.1	Users will be able to report errors/bugs or contact an admin about an issue	A	P	JR,SO,JK
NON-FUNCTIONAL REQUIREMENTS				
1- General				
1.1	Website is compatible with google chrome, firefox, IE-8, IE-9, IE-10 (Different browsers)		P	
1.2	Website is compatible with phones running ios or android. (Responsive Design)		P	
1.3	Website is offered for free as of now, and will have subscriptions later on and no ads as of now.		P	
1.4	The client has a limited / minimal budget for operational costs		P	
2- Performance				
2.1	Listings must load and be displayed within two seconds of user searching for a list		P	
2.2	Logging into the website must take less than two seconds		P	
3- Scalability				
3.1	The system should allow new listings to be added or removed from the database		P	
4- Capacity				
4.1	The server should be able to store all data and media for future listings		P	
5-Availability				
5.1	The website should be available to all users with a viable internet connection		P	
5.2	The database/server should be available to the admin at all times		P	
6-Reliability				
6.1	The server should have full error handling to avoid a crash		P	
7-Maintainability				
7.1	The system should be easily maintained and shouldn't require any technical knowledge to keep it running		P	
8-Security				
8.1	The media should be stored securely and only accessible to users of the website through the site on request,			
8.2	The admin shall be granted a separate and secure portal to be able to access the database/server		P	
8.3	The server must not allow outside access to the server and any media content stored on the server		P	

9-Regulatory				
9.1	Users must only need email, phone number, and password to register an account		P	
10-Data Integrity				
10.1	The device browser should maintain the integrity of persistent data		P	
10.2	The server should maintain the integrity of persistent data		P	
11-Usability				
11.1	The website should be easily usable by users with minimal technical training or knowledge		P	
12-Recoverability				
12.1	The server shall maintain the integrity of persistent data		P	
12.2	The site shall recover from all errors and exceptions without crashing		P	

User Test Report

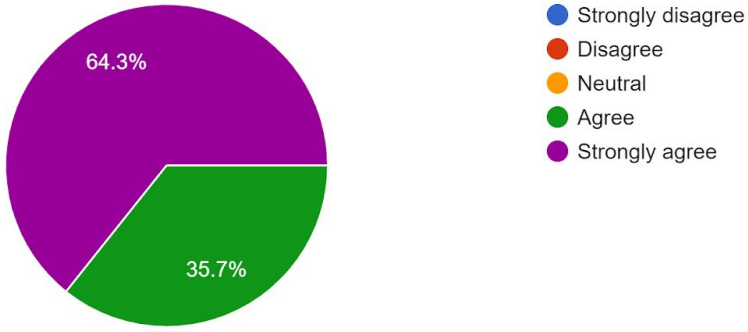
I found the U.I. visually appealing

14 responses



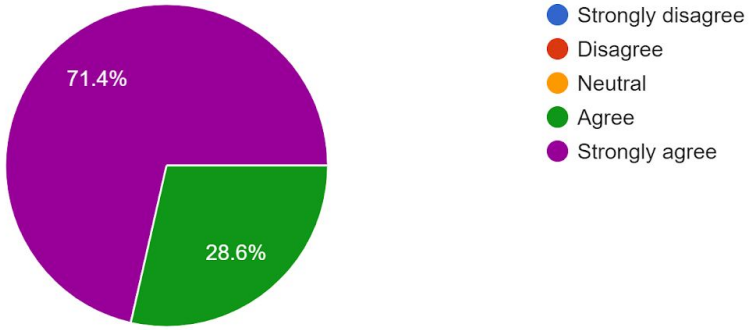
I found the U.I. easy to navigate

14 responses



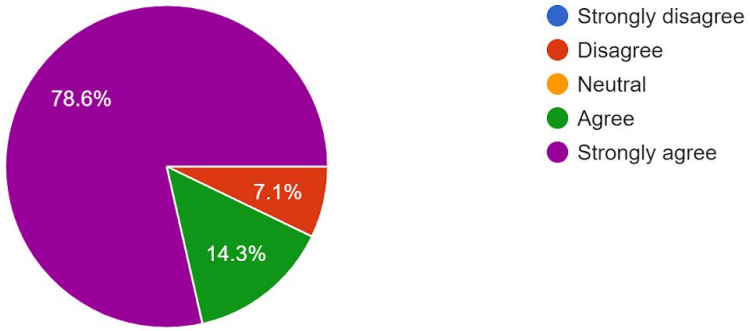
Pages loaded relatively quickly

14 responses



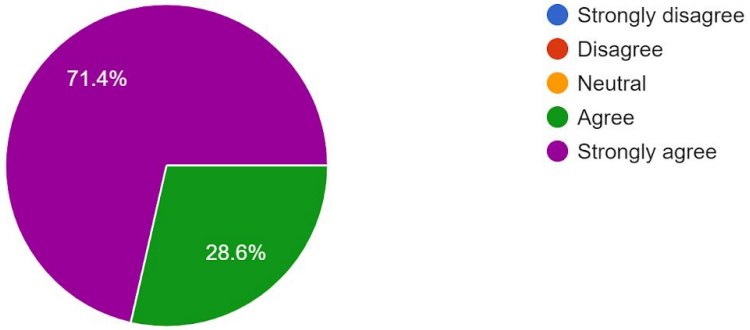
Logging in was quick

14 responses



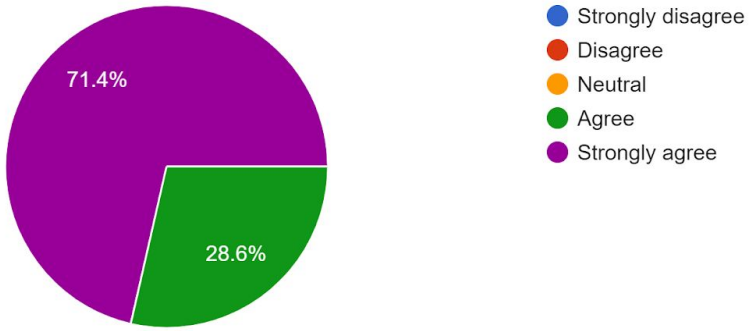
Registration was quick and straightforward

14 responses



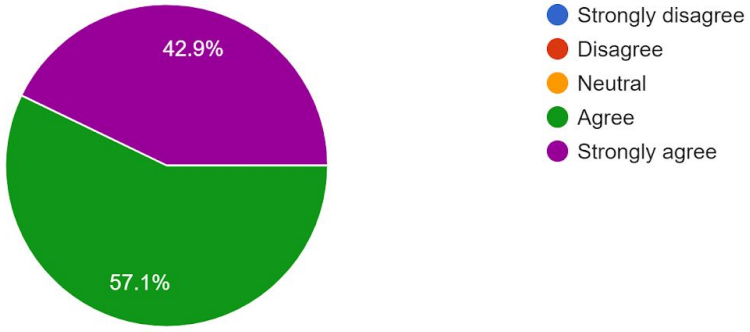
The system was quick and responsive

14 responses



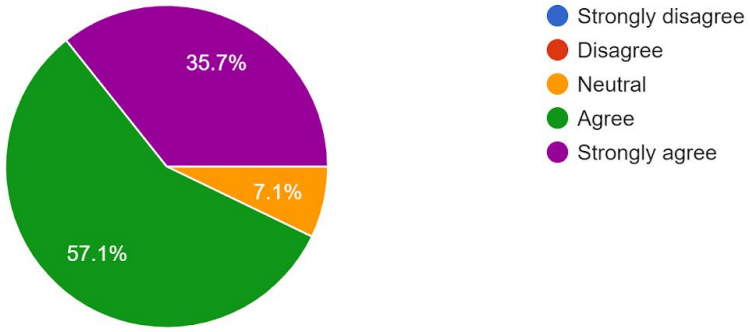
I would use this website in my daily life

14 responses



I would recommend this website to my friends and family

14 responses



Discussion

Below is the analysis to the feedback

- To start with, our user testing scenarios were very general and straightforward. We wanted users to have an easy and simple experience so that customers will have an easy time navigating and using functionality on the site.
- Most of users commented to us that our website looked clean and professional. We were happy with this result, as it was what our clients wanted.

Website Testing

Front-end Testing with selenium

- Testing of the website was done with a tool called Selenium and Selenium IDE. Selenium IDE works as Chrome and Firefox add-on that will do simple record-and-playback of interactions with the browser.It creates quick bug reproduction scripts, and creates scripts to aid in automation-aided exploratory testing.It can be found at http://www.seleniumhq.org/ . Inside our test folder, we have located test scripts that can be downloaded and imported directly into the selenium IDE. Once imported into selenium IDE, it will test all of the major components of the website and test all forms ,buttons, etc.

Database and PHP testing with PHPUnit

- Testing the database, a PHP framework called PHPUnit was used. Inside our test folder, databaseAPITest.php contains all the test scripts we did for each functions in databaseAPI.php. In order to test the database functions, it is crucial that each test starts with a clean database.

Reflection & Conclusion

Lessons Learned

This website developing project was very entertaining and helpful for us. Since most of us want to get a job as website developers when we graduate, we put in extra efforts and made sure we were trying our best. Thus, we learned many new technologies that we did not know in previous projects (COSC 304 & COSC 360). We learned how to work with frameworks, especially bootstrap. Our client wanted our website to be as clean as possible, so using bootstrap did help us a lot. We also learned that having a modular project structure helps tremendously in keeping things organized, testing, and editing parts of the project. Testing early on also helps a lot in knowing what parts of the code are causing issues.

What Would Be Done Differently

We already had background knowledge of website development from COSC 304 & COSC 360, but we were still fairly new to it and felt like development process could be better. Since, the project was done in such a short period of time, we felt like not enough documentation was done in the documentation phase. We were not able to separate our work clearly. It would also

be better to have the client define the exact requirements and wants better so that feature-creep is minimized.

Product Delivery

Code Location (Github)

https://github.com/ubco-cosc499-summer2019/group-5-web-marketing-platform-group-5-web-marketing-portal

Installation

- Github src files will need to be cloned to new server
- Database needs to be setup with DDL located at “src/database/Database DDL.sql”

Documentation

- All our documentation can be found in “docs” folder within the project repository.
- Typora was used to format .md files (logs).

Features Remaining & Future Work / Deficiencies

Below is a list of features not completed due to time constraints.

- Map search
- Editing listings or account details
- Disable/deleting listing and account by user
- Contacting landlord
- Bookmarking listing
- Phone number validation
- Search filter
- Automated search agent