VALIDATION DOCUMENT: APARTMENT RENT BIDDING SYSTEM

PROJECT INFORMATION

Project Name: Apartment Rent Bidding System
Submission Date: November 29, 2022
VD Version: 1.0
Client: Damen Tomassi

PROJECT TEAM

Team Member	**Role**	**Document Responsibilities**
Emma Dougherty	Development Team	Use Case
Alizsa Johnson	Development Team	Project Scope and Definition; Revision History
Sara Torres	Development Team	Validation Testing (Requirement Traceability)
Lucas Adams	Scrum Master	Validation Testing (Requirement Traceability)
Gregory Zacharko	Product Owner	Common Terms and Definitions Within Project Scope and Definition Section; Proofread VD

PROJECT SCOPE AND DEFINITION

The Apartment Rent Bidding System is a Web application that allows apartment administrators to auction off apartment units to potential renters.

The administrators create apartment buildings, populate those buildings with apartment units, and set the minimum going price and bid rate. The apartment administrator gives the bidders the apartment unit ID. Then the bidders start placing bids and continue bidding until the auction ends. At the end of the auction, a winner is selected and both the winning bidder and the apartment administrator are notified.

The scope of the Apartment Rent Bidding System is to create a Web application where apartment administrators can hold auctions for apartment units and invite bidders to bid on the rent.

<u>For Administrators</u>

- Allow administrators to sign up for a special user account that allows them to create apartment buildings and display available units.
    - Once an administrator creates an apartment unit, it is assigned a unique identifier that is visible to the administrator. This identifier allows approved bidders to access the auction.
- Allow administrators to set the minimum bid and minimum bidding increment.
- Allow administrators to add apartment details such as lease length and amenities.

<u>For Bidders</u>

- Allow bidders to sign up for an account that lets them participate in auctions.
- Allow bidders to bid on apartment units.
- Allow bidders to use an auto bidding feature, where they set the max amount they'd like to spend and the bidding system will bid on the users behalf until no other competing bids are placed or until the bidders max price is reached.
- Allow bidders to be notified when they've been outbid and when they've won the auction.

<u>Deliverables</u>

This project will yield a fully functional Web application that allows apartment administrators to auction off apartment units and allows bidders to access and participate in the auctions.

<u>Milestones</u>

- Sprint 0:
    - Decide what kind of pages need to be present on the web application and create the GUI wireframes in Figma.
    - Research the features of Bubble.io.
- Sprint 1:
    - Recreate the Figma diagrams in Bubble.io and begin adding functionality to the web pages.

    - Set up the database for the apartment listing details so that it stores admin input and displays it on the web page.
    - Verify that administrators can create listings.
- Sprint 2:
    - Set up additional databases that will hold the account information for the bidders and administrators and verify that both user types are able to sign up and log in.
    - Create the bidding feature that will allow bidders to participate in the auction.
- Sprint 3:
    - Create the auto-bidding feature and decide how both bidders and the administrator will be notified of significant events related to the auction.
    - Look into using a plugin to send out email updates to the auction participants.
- Sprint 4:
    - Begin bug testing the web application. All of the key features will be implemented so the next step is to verify that the application is working as intended.
    - Begin implementing stretch goals.
- Sprint 5:
    - Test any added features and confirm that they are working properly.

COMMON TERMS AND DEFINITIONS:

1. SPONSOR: The owner/stakeholder of the product.
2. PROFESSOR: Third-party overseer of the project and the team developing it.
3. ADMINISTRATOR: A person that creates a listing for an apartment unit using the product.
4. BIDDER: A person that uses the product to bid on an open and available apartment unit.
5. APARTMENT UNIT ID: A randomized, unique sequence of letters and numbers that bidders use to access the specific apartment listing they want to bid on.
6. BUBBLE.IO: A software engineering platform that allows developers to quickly build high-quality, fully functioning, powerful desktop/Web and mobile applications without having to type out thousands of lines of code.
7. ELEMENT: A component on the website/Web application that the user can interact with (e.g. buttons, links, text input boxes) or that displays information to the user (e.g. dynamic text displays).
8. ELEMENT PALETTE: The place where a developer can pick an element to drag-and-drop onto the page. It is where all possible elements are listed.

9. WORKFLOW: The different user interactions that can occur between the administrators and bidders, and the website/Web application. It is the sequence/flow of events and interactions when an administrator and/or bidder uses the product.
10. FIGMA: A software platform that allows developers to create a wireframe for their project/product's GUI.

REVISION HISTORY

Sprint	**Changes in Scope, Design, or Programming**
0-1	● Developed team communication strategies ● Created wireframes for each page of the web app
1-2	● Learned Bubble.io interface and tools ● Translated wireframes into site pages on Bubble.io
2-3	● Added additional fields to the "Create Listing" popup per the sponsor's request ● Decided on using two different databases to hold user and apartment information ● Downsized the number of site pages required
3-4	● Added an "Amenities" field ● Modified bid and auto bid feature to work properly under all conditions
4-5	● Revamped site design and made it more visually appealing and user friendly ● Removed the "Amenities" field and combined it with the listing's description ● Removed "Create Listing" popup and replaced it with a "Create Listing" site page

USE CASE

Step	**Use Case**	**Test Steps**	**Test Data**	**Expected Result**

1	Create Account	1. Navigate to index page (homepage)
		2. Click “Sign Up” button
		3. Enter credentials on popup
		4. Select whether you are creating listings or bidding on listings on popup
		5. Click “Create Account” button on popup
			Id= user@gmail.com
			PW= password
				Account Creation Successful

2	Log In	1. Navigate to index page (homepage) 2. Click “Log In” button 3. Enter credentials 4. Click “Log In” button on popup 5. Brought to dashboard	Original Log In Credentials	Log In Successful
3	Add Listing (Bidder)	1. On dashboard, click “Add Listing” button 2. Enter listing ID given by realtor on popup 3. Click “Add Listing” button on popup 4. Listing displayed on dashboard	Id = 1234	Listing Added

4	Bid on Listing	1. On dashboard, click “View” button on listing preview 2. Brought to listing page 3. Click on “Bid” button to bid on apartment with the set increment 4. Top bid updated	Dependent on Bid Amount Set by Admin	Bid Entered Successfully
5	Autobid on Listing	1. On dashboard, click “View” button on listing preview 2. Brought to listing page 3. Click on “Autobid” button on listing page 4. Enter max bid amount on popup 5. Click “Set Autobid” button	Max bid = 6000	Autobid Entered Successfully
6	Create Listing (Admin)	1. On dashboard, click “Create Listing” button 2. Brought to create listing page 3. Enter listing information in related fields 4. Enter starting bid 5. Enter end date for bidding 6. Upload photos of listing 7. Select whether bidders can see	Dependent on Admin Input	Listing Added Successfully

		your contact information 8. Click “Create Listing” button 9. Listing should be seen on dashboard		
7	Reset Password	1. Navigate to index page (homepage) 2. Click “Log In” button 3. Click “Forgot Password?” button 4. Enter email in forgot password popup 5. Navigate to email 6. Click on link sent by apartment bidding system (Bid + Rent) 7. Enter new password on page 8. Click “Reset Password” button	User’s Original Email Credentials	Password Reset Successfully

VALIDATION TESTING (REQUIREMENT TRACEABILITY)

Requirement Validation	**Metrics**	**Action Items**
System and OS Environments	Ability to run a modern Internet browser. Ability to have access to the Internet.	Confirm access to the Internet and that it is working. Download a better Internet browser or one that allows the page to function.
Hardware Requirements	For Windows: - Memory: 2 GB minimum - Screen Resolution: 1280x1024 or larger - Windows 8 or later - Internet connection required	For Windows: - Acquire Windows 8 or later - Troubleshoot Internet Issues if the Internet is not working properly - Acquire more memory space if there is less than the minimum
Software Requirements	All that is required is a modern Internet browser (Google Chrome, Firefox, etc.)	- Clear browser cache - Disable firewall and/or antivirus - Go to the website on a different device or network
Installation and Configuration	No installation required. Configuration includes creation of user profiles.	Contact support if the creation of user profiles is not possible.

Network Requirements	Internet connection is required to connect to the website.	Troubleshoot the Internet by restarting the equipment, connect with an Ethernet cable, check for outages in your area, etc.
Code Testing and Documentation	1. No code and no code comments are available since it was made in Bubble.io 2. We use the Preview feature in Bubble.io, where we can test multiple times how the page works and how pages interact with each other 3. First, by making sure the workflow doesn't provide any errors. Then, with the use of Preview we can check how it works 4. If something has an error it will simply not work, or it won’t work accordingly. Also, we will have notifications that will show us where the application is not working correctly 5. Test cases	Since we worked with no code, the application functionality is within the workflow of the application. Therefore, making sure that they work, we can assure the application is working.

Key User Operations	Users are able to click multiple buttons which will allow them to view specific listings (if they have the unique code), for which they are able to bid and autobid. They are also able to create a profile and add multiple listings to their dashboard if they have multiple codes. Admins can create a unique profile specific for admins which will allow them to create listings and share it with users giving them a unique code. They also will be able to edit these listings in the listing page.	If users and admins are not able to create profiles or listings, or press a button for a specific action. We will notice and fix the issue.
Data Management	We have fields that help us collect, organize, and protect the data so it can be analyzed.	Bubble.io makes it easy to organize and store data, which will make it hard for a data loss or leak to happen.
Database	Bubble.io can easily keep track of the data input. We have fields for the Users, Admin, Listings, etc. Every profile is secured.	When creating a profile or listings, the data is automatically stored and saved. In case this didn't happen the user won't be able to finalize the process of creating a profile or listing.

Data Inputs (All inputs that the software product will receive. Includes specification of ranges, limits, defaults, response to illegal inputs, etc.)	User Profile Inputs: ● Account Email ● Account Password ● User’s Name ● Phone Number ● Boolean for creating listings or bidding on them Create Listing Inputs: ● Title of Listing ● Address of Listing ● Description of Listing ● (optional) Complex name (such as apartment) ● (optional) Unit Number ● Bedrooms ● Bathrooms ● Starting Bid ● Bidding Increment ● Final Bid Date ● Images for the listing page ● Boolean for sharing listing creator’s contact information or not Bidding: ● Highest Autobid ● When the user raises the bid	Users will be informed if the action made was wrong. With case scenarios, we made sure that every feature and input works properly.

Data Outputs (All outputs that the software product will produce. Includes data formats, screen presentations, storage medium, printouts, generation of documents, etc.)	● A unique code for each listing for connections ● A listing page to present the listing to users ● Final bidder and bid price of a listing after bidding ends	In case the admin does not receive a unique code, we check that the workflow for random code generation works.
Interface and Usability Testing	Bubble.io has a very useful feature: Preview. This allows us to test different case scenarios and to test the usability without having external users, since we can create fake user/admin profiles.	The way to fix this, if the interface doesn't work, is through Bubble.io's workflows and design features.
Performance/Responsiveness	The website performance must allow users/admins to create a profile. Must allow admins to create listings, receive a unique code, and put a starting bid price. Must allow users to create a profile, add the unique code they received, bid and outbid in any listing they can have access to.	
Error Checking Management	User error can be found in two places: when inputting credentials and when bidding. ● When inputting credentials, for	

	instance inputting a phone number, if the input is not valid when trying to submit a pop-up will inform you of the input not being valid and you will not be able to proceed until it is ● When bidding, if the user tries to input a bid that is lower than the current highest bid, a pop-up window will inform them that their bid must be higher	
Peer Testing Procedures	Peers will: ● Create multiple accounts to verify that the all features are working as intended, e.g.: workflow logic catches invalid input, admins cannot bid, and bidders cannot create listings ● Bid on sample auctions by inputting both valid and invalid bids to test the bidding logic. Bids lower than and equal to	Any errors caught will need to be corrected by viewing the workflow for the problematic feature and the logic will need to be edited and then tested again.

	the top bid should be rejected and higher bids need to be displayed as they come through ● Test the auto bidding feature by entering an auto bid that is lower, higher, and equal to both the highest bid and the highest auto bid. This test and the previous one will simulate a “real time” auction, which allows us to see what happens when multiple people are using the application at the same time	
Client Review and Procedures	The presentation will include demonstrations of every function. 1. Create an admin account 2. Create a listing 3. View the listing and demonstrate the edit function 4. Sign out, and create a user profile 5. Connect the listing using a code and bid on the listing	

	6. Sign out and create a second user, connect the listing, and bid on the listing using autobid 7. Sign in as the previous user and bid, but get outbid by the previous user's autobid	