Use Cases

for

<House Buying Guide>

Version 1.0 approved

Prepared by <Corliss Lim>

<Meatballs>

<28/08/2020>

Revision History

Name	Date	Reason For Changes	Version

Copyright © 1999 by Karl E. Wiegers. Permission is granted to use, modify, and distribute this document.

Use Case ID:	1		
Use Case Name:	Search historical data		
Created By:	Duan Kai	Last Updated By:	Duan Kai
Date Created:	28082020	Date Last Updated:	28082020

Actor:	**User**
Description:	Retrieve resale/private sale data according to user-defined search criteria (residential type, location, type of housing, floor level, lease information, square footage, price range (total), price range ($psf), sale type, project name), with parameters displayed according to user's intended purpose
Preconditions:	N.A.
Postconditions:	1. System displays price of housing according to criteria 2. If user is logged in, record search criteria in database and associate search parameters with their account
Priority:	Highest
Frequency of Use:	5000 times a week
Flow of Events:	1. **User** selects the option to *search historical data,* and may select to search as a buyer or a seller 2. If buyer is selected, System displays parameters of sale type, residential type, location, type of housing, floor level, lease information, square footage, price range (total), and price range ($psf) 3. If seller is selected, System displays parameters of project name, residential type, location, type of housing, floor level, lease information, square footage, price range (total), and price range ($psf) 4. If neither is selected, System displays parameters of residential type, location, type of housing, floor level, lease information, square footage, price range (total), and price range ($psf) 5. System requests the input of criteria to filter historical data 6. **User** enters one, several or all criteria to filter historical data 7. System searches through the **Server database** for historical data matching input criteria 8. If matching historical data is retrieved, System displays the data to **User** 9. If **User** is logged in, System runs *store search/request history*
Alternative Flows:	1.AC.S4: If System cannot find search criteria selected in database 1. System prompts user with an error message, and prompts user to enter another set of criteria 2. System returns to step 2
Exceptions:	1.EX.S4: If **Server database** is inaccessible or empty 1. System prompts user with an error message "server offline" 2. System returns **User** to the homepage

Includes:	*Store search/request history*
Special Requirements:	1. **User** may further filter search results by refining previous search criteria through returning to step 3
Assumptions:	**User** uses English
Notes and Issues:	N.A.

Use Case ID:	2		
Use Case Name:	Request financing guide		
Created By:	Corliss	Last Updated By:	Duan Kai
Date Created:	28082020	Date Last Updated:	28082020

Actor:	**User**
Description:	System suggests a financing plan based on user input parameters (price of housing, current available funds, desired monthly repayment, desired repayment duration)
Preconditions:	N.A.
Postconditions:	1. System financial guide according to parameters 2. If user is logged in, record search criteria in **server database** and associate search parameters with their account
Priority:	Medium
Frequency of Use:	500 times a week
Flow of Events:	1. **User** selects option to *request financing guide* 2. System requests the input of at least three out of the four parameters 3. **User** enters at least three out of the four parameters 4. System passes parameters to **Server Database** 5. **Server Database** returns financing guide (loan amount, unentered parameter) to system 6. System displays financing guide to **User** 7. If **User** is logged in, System runs *store search/request history*
Alternative Flows:	2.AC.S3: User enters less than three out of the four parameters 1. System prompts user with error message “insufficient information for calculation”
Exceptions:	2.EX.S4: If **Server database** is inaccessible or empty 1. System displays error message “server offline” 2. System returns **User** to the homepage
Includes:	*Store search/request history*
Special Requirements:	N.A.
Assumptions:	**User** uses English
Notes and Issues:	N.A.

Use Case ID:	3		
Use Case Name:	Request price prediction		
Created By:	Corliss	Last Updated By:	
Date Created:	28082020	Date Last Updated:	

Actor:	**User**
Description:	System returns a predicted price for a housing unit in the future
Preconditions:	1. **User** makes a successful query to the system, containing at least one parameter among location, housing type, square area, floor level, lease information
Postconditions:	1. Price prediction is displayed to the **User** 2. If user is logged in, record query and result in the database
Priority:	High
Frequency of Use:	2000 times a week
Flow of Events:	1. **User** makes a query to the system 2. System verifies that an appropriate query has been made by the **User** 3. System returns a price prediction 4. If **User** is logged in, System runs *store search/request history*
Alternative Flows:	3.AC.S2: Invalid query made by user 1. System highlights error and prompts for another query
Exceptions:	3.EX.S3: If **Server database** is unable to predict a price 1. System displays error message “server error” 2. System returns **User** to the home page. 3.EX.S4: If **Server database** is unable to record price prediction request 1. System displays an error message “unable to record query” 2. System returns **User** to the home page.
Includes:	*Store search/request history, Login*
Special Requirements:	1. The system only outputs a predicted price if it has a confidence score of at least 80%. 2. The system will return an error if the confidence score of the predicted price is below 80%.
Assumptions:	**User** gives a correct query that is within valid range of search parameters
Notes and Issues:	N.A.

Use Case ID:	4		
Use Case Name:	View search history		
Created By:	Corliss	Last Updated By:	
Date Created:	28082020	Date Last Updated:	

Actor:	**User**
Description:	**User** views history of all price prediction queries, request of financial planning and historical data search request
Preconditions:	1. **User** must be *logged in* to the system
Postconditions:	1. System displays search history of user
Priority:	Low
Frequency of Use:	250 times a week
Flow of Events:	1. **User** selects option to view search history 2. System pulls user search history data from **Server database** 3. System displays search result to **User**
Alternative Flows:	4.AC.S1: **User** is not logged in 1. System displays an error message of “Function not available. Please log in to continue” 2. System prompts the **User** to log in to an account. 4.AC.S2: **User** does not have a search history 1. System displays an error message of “No search history for <username> found”. 2. System returns to homepage.
Exceptions:	4.EX.S2: If **Server database** is inaccessible or empty 1. System displays error message“server offline” 2. System returns **User** to the home page. 4.EX.S2: If **Server database** is unable to find search history of user 1. System displays an error message “server error” 2. System returns **User** to the home page,
Includes:	*Log in*
Special Requirements:	1. System displays a default display of search results based on date. 2. System allows **User** to filter search results based on categories of request price prediction, search historical data and request a financial plan.
Assumptions:	**User** has a search history
Notes and Issues:	N.A.

Use Case ID:	5		
Use Case Name:	Store search/request history		
Created By:	Duan Kai	Last Updated By:	Duan Kai
Date Created:	28082020	Date Last Updated:	28082020

Actor:	**Server database**
Description:	Store search/request history of user and associate it with the user such that they may repeat the same request in the future without reentering every field.
Preconditions:	1. **User** is logged in 2. Either *request price prediction*, *request financing guide* or *search historical data* is made by **User**
Postconditions:	1. If *request price prediction* was made by **User**, an entry containing the date the query was made, price predicted and the parameters input by the **User** is stored in the database 2. If *request financing guide* was made by **User**, an entry containing the date the query was made, the parameters input by the **User**, and the financing plan output values is stored in the database 3. If *search historical data* was made by **User**, an entry containing the date the query was made and the parameters input by the **User** is stored in the database
Priority:	Low
Frequency of Use:	2000 times a week
Flow of Events:	1. **User** has made a *search historical data, request price prediction,* or *request financing guide* request to the System. 2. System passes search/request parameters to **Server Database** 3. **Server Database** stores search/request parameters, and associates the parameters with the user
Alternative Flows:	N.A.
Exceptions:	5.EX.S4: If **Server database** is inaccessible or empty 1. System displays an error message “server offline” 2. System returns **User** to the homepage
Includes:	N.A.
Special Requirements:	All date entries are recorded in dd/MM/yyyy HH:mm:ss format
Assumptions:	**User** uses English
Notes and Issues:	N.A.

Use Case ID:	6		
Use Case Name:	Log in		
Created By:	Corliss	Last Updated By:	Duan Kai
Date Created:	28082020	Date Last Updated:	28082020

Actor:	**User**
Description:	User log into system
Preconditions:	N.A.
Postconditions:	1. **User** state changed from guest to logged in 2. **Server** will run *store search/request history* on future searches & requests 3. **User** is able to run *view search history*
Priority:	Medium
Frequency of Use:	2000 times a week
Flow of Events:	1. **User** selects option to *log in* 2. System requests the input of a email and password 3. **User** enters email and password 4. System verifies that email and password are in the **Server Database** 5. If the email and password are verified, System logs in the **User** and displays a success message
Alternative Flows:	6.AC.S2: **User** has forgotten password 1. **User** selects option to reset password 2. System moves user to *reset password* 6.AC.S2: **User** does not have an account 1. **User** selects option to signup 2. System moves user to *signup* 6.AC.S5: Email and password are not verified 1. System prompts user with error message “Login credentials incorrect” 2. System returns to step 2
Exceptions:	6.EX.S4: If **Server database** is inaccessible or empty 3. System displays an error message “server offline” 4. System returns **User** to the homepage, with error message “server offline”
Includes:	*Reset Password*, *Signup*
Special Requirements:	N.A.
Assumptions:	**User** uses English
Notes and Issues:	N.A.

Use Case ID:	7		
Use Case Name:	Signup		
Created By:	Duan Kai	Last Updated By:	Duan Kai
Date Created:	28082020	Date Last Updated:	28082020

Actor:	**User**
Description:	Allows **User** to sign up for an account using an email and password
Preconditions:	N.A.
Postconditions:	1. **System Database** has created a new user with email, name and password parameters
Priority:	Low
Frequency of Use:	2000 times a week
Flow of Events:	1. **User** selects option to *signup* 2. System requests the input of an email, a name and a password 3. **User** enters a valid email, a name and a password 4. System passes the email and password to the **Server Database** 5. If the email is not in use, **Server Database** creates a new user with the input email and password 6. System displays a success message “Account created! Please login from the homepage” 7. System returns **User** to the homepage
Alternative Flows:	7.AC.S4 : If email entered already exists in the **Server database** 1. System displays error message “Email already in use” 2. System returns to step 2 7.EX,S10: Password is not strong enough 1. System displays error “password not strong enough” 2. System returns to step 8
Exceptions:	7.EX.S4: If **Server database** is inaccessible or empty 1. System displays an error message “server offline” 2. System returns **User** to the homepage
Includes:	N.A.
Special Requirements:	System must verify that a valid email has been provided before approving the sign up System must verify that a valid password has been provided before approving the sign up. A valid password must be between 8 to 21 characters and have no spaces or special characters
Assumptions:	**User** uses English
Notes and Issues:	N.A.

Use Case ID:	8		
Use Case Name:	Reset password		
Created By:	Duan Kai	Last Updated By:	Duan Kai
Date Created:	28082020	Date Last Updated:	28082020

Actor:	**User**
Description:	Allows **User** to reset password for their account
Preconditions:	1. **User** already has an account
Postconditions:	1. **Server database** is updated with new password
Priority:	Low
Frequency of Use:	2000 times a week
Flow of Events:	1. **User** selects option to reset password 2. System requests the input of an email 3. **User** enters email 4. System verifies that email is in the **Server Database** 5. If the email is valid, System displays a success message "we have sent an email to your provided email address with password reset information" 6. System sends an email to provided email address with a link to reset user password 7. **User** opens email and clicks on provided link to reset password 8. System requests the input of a new password 9. **User** enters a new password 10. System passes password to **Server Database** 11. **Server Database** stores new password associated with the email 12. System displays a success message "password reset! please login from the homepage" 13. System returns **User** to the homepage
Alternative Flows:	8.AC.S4: Email is not in the **Server Database** 1. System displays error "email not found" 2. System returns to step 2. 8.EX,S10: Password is not strong enough 3. System displays error "password not strong enough" 4. System returns to step 8
Exceptions:	8.EX.S4: If **Server database** is unable to verify email 1. System prompts user with an error message "server offline" 2. System returns **User** to the homepage 8.EX.S11: If **Server database** is unable to record new password 1. System prompts user with an error message "server offline" 2. System returns **User** to the homepage
Includes:	N.A.
Special Requirements:	System must verify that a valid password has been provided before approving the password reset.

	A valid password must be between 8 to 21 characters and have no spaces or special characters **User** will have to enter new password twice and the system will only approve password reset request if the entered passwords match
Assumptions:	**User** uses English
Notes and Issues:	N.A.