Simon Fraser University

Realtorest - Iteration 3

Requirements and Specification Document

Project Group 6

CMPT 276: Introduction to Software Engineering

Kevin Cheng

Nam Nguyen

Drishty Dhiman

Haramrit Toor

Malaika Qureshi

Requirements and Specification Document

Project Name:	Realtorest	
Client Contact:	Aman Dhiman	250-588-0739
Project Due Dates:	Pre-production	February 25, 2024
	Iteration 1	March 12, 2024
	Iteration 2	March 26, 2024
	Iteration 3	April 9, 2024
Documentation Submission Dates:	Project Proposal	February 25, 2024
	Iteration 1	March 12, 2024
	Iteration 2	March 26, 2024
	Iteration 3	April 9, 2024

Project Abstract

Realtorest is a real estate web application that is created for Aman, a realtor located in British Columbia. The application displays a list of properties that can be sorted based on price, location and room features, and the available map feature can assist users to explore properties in different locations. From a general users' perspective, users can create an account, manage their basic information, control their property choices with features such as 'Save as Favorite', and connect with Aman to discuss or set up an appointment. The application is specifically tailored for our client Aman, who is the middleman that helps property sellers and buyers connect with each other. This application focuses on helping Aman upload and display information of properties that are being sold and help potential buyers who are interested in those properties to make the purchase.

Client

Aman is a realtor in British Columbia who does not have a personal website for his real estate business, and due to the competitiveness in the industry, a personal website can benefit Aman a lot by helping him connect with potential clients and widening his business scope. Hence, the web application Realtorest is created to help Aman connect to property buyers and sellers. The description will outline two types of users that will use this software: Aman, the project's client, who will be the admin of the website. And the general customers, who are home buyers, real estate investors, renters, and real estate professionals. Detailed below are the high-level descriptions of the features that Realtorest has to offer.

Competitive Analysis

There are many personalized websites tailored for realtors similar to Aman, hence it is very competitive for the project to have its own distinctive features that stand out from the rest of the other websites. However, Realtorest is different from many other realtor websites in which it does not have any advertisements, because the project is focused on the client, rather than profit like the others. In addition, under UI perspective, most of the realtors websites that the team encountered are usually very hard to navigate and not user friendly, since there are many website features and interactions that all appear at once. Hence, to avoid this problem, Realtorest will focus on the customer interaction experience by creating a minimalist layout that can help the user to navigate with ease, while also providing sufficient features.

User Stories

Introduction: Realtorest is a website that helps the client, who is a realtor, connects with his potential customers; hence the actors involved will be the administrator, registered users, and unregistered users. The administrator will be Aman, our client who can add new listings, manage existing listings, and contact interested customers. Registered users can view listings, contact Aman for more information, and bookmark their favorite properties. Unregistered users have the ability to browse the listing and contact Aman for more information.

Site Navigation

Actor: Alice is an unregistered user visiting the website for the first time. Alice wants to look around and see what information is provided by the website.

Precondition:

- Alice lands on the home page, where she can navigate to different pages of Realtorest using the navigation bar on the top of the page.

Action:

- Clicking on “Realtorest” will redirect Alice to the homepage.
- Clicking on “About Us” will cause a drop-down tab to appear from the top containing Aman’s name, title, phone number, clickable social media links, and a brief introduction of Aman.
- Clicking on the “Listing” link on the navigation bar will redirect Alice to the property listings page.
- Clicking on “Mortgage Calculator” on the navigation bar will direct Alice to a page where she can see what mortgage payments may look like for a property.
- Clicking on “Favorites” will redirect Alice to the login page as that feature is only for registered users.
- Clicking on “Sign In” will lead Alice to the login page and since she doesn’t have an account, clicking the “Register Here” link on the login form will lead her to the register page to make one.

Acceptance Criterias:

- The home page needs to be able to load while the website is up and running.
- When the user clicks on a button, they need to be either redirected to the corresponding page, or the relevant information needs to be displayed.

Tests:

- RealtorestApplicationTests.java > testContextLoads
    - The server can be hosted up and running.

- ControllersTests.java > testShowHomePage
    - The home page can be displayed.

Iteration: 1
Story Points: 1

Property Listings

Actor: Joan is a new user, who is using Realtorest for the first time and just wants to see what the website has to offer.
Preconditions:

- Joan clicks on the “Listings” or “Explore Our Listings” and gets redirected to the listing page.

Actions:

- Joan can see all available properties on the listing page, such that only 9 properties will be displayed at a time. She can use the pagination bar below the list to navigate to see more properties.

Acceptance Criterias:

- The listing page must only show 9 properties at a time and navigating through the pagination will display different properties.

Test:

- ControllersTests.java > testShowPropertyListingsPage
    - The property listing page can be displayed.

Iteration: 1
Story Points: 2

Property Listings - Viewing Details

Actor: Jeremy is on the property listing page, and he wants to explore some potential properties that he will be invested in the future.
Preconditions:

- He is interested in seeing more information about a property near his workplace and decides to click on the figcaption under the property’s image carousel.

Actions:

- A modal window will popup and display a bigger images carousel of the property. The carousel will automatically change the picture every few seconds when Jeremy is not hovering his cursor onto the picture.
- A detailed description of the property and the realtor’s information will also be displayed under the carousel.

Acceptance Criterias:

- A popup modal window will be displayed after the user clicks on the property figcaption.
- The modal window will display the correct information of the property, and it should automatically change the carousel’s picture every few seconds if the user is not hovering their cursor on the picture.
- Clicking outside of the modal window will elegantly close the window.

Test:

- PropertyControllerTest.java > testContextLoadsPropertyListingsPage

    - The property listings page can be loaded.
- PropertyControllerTest.java > testGetProperties_NoParams_ShouldReturnAllProperties
    - If no filters are set, the property listings page should show all properties.

Iteration: 2
Story Points: 3

Property Listings - Viewing Google Maps

Actor: Marko is browsing some listed properties, and he wants to have a glimpse of whereabout the property is located on the map.
Preconditions:

- Marko clicks on the property figure caption.

Actions:

- A detailed description modal box of the property will appear, and after Marko scrolls to near the bottom, he will see a snapshot of the location through Google Maps.
- Marko can interact with the map, such as zoom in or out or simply drag the map to view the nearby locations.
- Clicking on the satellite map located at the bottom left of the map will change the map into satellite view. In addition, clicking on 'Directions' located on the top left of the map will navigate Marko to Google Maps, where he can have the direction from his current location to the property location.

Acceptance Criterias:

- The snapshot displays the correct location of the property, and Marko can interact with the maps.

Iteration: 3
Story Points: 2

Property Listings - Filtering and Sorting

Actors: Emily, a young professional on a budget, looking for an affordable home with 3 bedrooms.
Preconditions:

- Emily uses the search filter on the listing page to filter out properties with 3 bedrooms and uses the bar to sort the property price from low to high, and she then clicks on the "Apply Filters" button.

Actions:

- The page presents the property that has 3 bedrooms.

Acceptance Criterias:

- The page displays the property that matches with the user's filter criterias. In other words, none applicable properties to the filter search will be hidden.

Iteration: 2
Story Points: 2

Account - Registration

Actor: Bob is an unregistered user wanting to register for an account.

Precondition:

- From the home page, Bob clicks the Sign In button on the top right, then clicks “Register Here” to visit the registration page.
- Bob then fills out the registration form, and he clicks the “Register” button.

Action:

- The system will validate the information Bob has entered. If any of those conditions fails, the system will prevent the form from being submitted.
- The user has left one of the fields empty.
- The contents of the two password fields do not match.
- The password is not strong enough (minimum of 8 characters, and include uppercase, lowercase, numbers, and special characters).
- After all validations are passed, Bob's account will be created in the database, with his information stored.
- Bob will then be redirected to the home page.

Acceptance Criterias:

- Bob needs to be able to visit the registration page.
- The information Bob has entered needs to be validated.
- If the form validation has failed:
    - No account should be created in the database.
    - The user should not be redirected to the home page.
    - There should be visual prompts telling the user what went wrong.
- If the validation has passed, the account needs to be created in the database.
- The user then needs to be redirected back to the home page.

Tests:

- ControllersTests.java > testShowRegisterPage
    - The register page can be displayed.
- ControllersTests.java > testUserRegister
    - The user can register after entering all the correct information.
- ControllersTests.java > testUserRegister
    - The user can be redirected to the home page after a successful registration.

Iteration: 1

Story Points: 2

Account - Login and Logout

Actors: Shane is a registered user who wants to sign into his account.

Preconditions:

- From the home page, Shane clicks the Sign In button on the top right, which leads him to the login page.
- Shane already has an account on the website, and his account information is being stored in the database.
- Shane enters his credentials and clicks the “Sign In” button.
- After his session, Shane wants to sign out, so he clicks on the “Sign Out” button.

Actions:

- The system will validate the form:
    - If a field is empty, Shane will be prompted to fill in what is missing.
    - If the “Email” field is not in a correct format (e.g. username is being used), Shane will be prompted to fix this.
- Then, it will search the database with the email and password combination entered.
    - If no matching account can be found, Shane will not be able to login and will remain on the same page.
- If the entered email and password combination is found, the login information will be stored in Shane’s session cookie.
- Then, Shane will be redirected to the home page.
- After clicking on the “Sign Out” button:
    - Shane’s session will be invalidated, and he will be redirected to the home page.

Acceptance Criterias:

- Shane needs to be able to visit the registration page.
- The information Shane has entered needs to be validated.
- If invalid credentials are entered:
- Shane needs to not be able to login and no session information should be stored.
- Shane needs to stay on the same login page.
- If the correct credentials are entered, Shane needs to be logged in, and his browser now needs to contain the correct session information.
- Shane then needs to be redirected back to the home page, and can see “Sign Out” on the top right replacing the “Sign In” button.
- After signing out, Shane must be redirected to the home page. With his session ended, the “Sign In” button will reappear on the navigation bar.

Tests:

- ControllersTests.java > testShowLoginPage
    - The login page can be displayed.
- ControllersTests.java > testUserLogout
    - The user can be redirected to the home page after logging out.

Iteration: 1

Story Points: 2

Account - Changing User Settings

Actors: Bonnie wants to make some changes regarding her account’s information.

Preconditions:

- Bonnie is a Realtorest user, and after signing in, she wants to make some changes to her account.

Actions:

- Once Bonnie clicks on the ‘Settings’ tab on the navigation bar, she will be navigated to the user settings page, where she can see her account information: first and last name, username, email address, password, and toggle box for mailing subscription.
- Upon clicking on one of the information fields, Bonnie can modify her information, while the ‘Save’ button will be enabled.

- If Bonnie decides that she wants to subscribe to the mailing list, she will click to toggle the mailing subscription box.
- After Bonnie has changed her information and is happy with her new information, she will click the 'Save' button to save her new information to the database.

Acceptance Criterias:

- The user has an account and is in a session.
- The user clicks the 'Save' button to submit the form, so the new information will be saved.

Iteration: 3
Story Points: 2

Account - Password Reset

Actors: Tak is a user who is trying to login to his account.
Preconditions:

- He remembers the email he used but cannot remember the password to his account.

Actions:

- He clicks the "Forgot Password?" button on the login page which leads to a page that has two input fields, one that asks him to enter his email address and the second which asks him to enter it again for confirmation sake.
- Once Tak clicks submit, a mail is sent to his email with a link.
    - If the email fields aren't the same, an alert pops up saying that the emails don't match.
    - If the email does not have an account, the user is redirected to the register page.
- The link leads to a page that has two input fields to reset his password, one that asks him to enter the new password and the other asks him to enter it again to confirm.
- Once Tak enters his new password and clicks submit, he is redirected to the login page where he can sign in to his account with the new password.
    - If the password fields aren't the same, an alert pops up saying that the passwords don't match.

Acceptance Criterias:

- The user has an account.
- An email is sent to the user's mail to reset their password.
- The link in the email leads to a reset password page for the user.
- The user clicking 'Submit' on the reset password page redirects to the login page.
- The user is able to login to their account with their new password.

Tests:

- ControllersTests.java > testShowForgotPasswordPage
    - This test calls the showForgotPasswordPage method and checks that it returns the string "redirect:/", which would redirect the user to the home page.
- ControllersTests.java > testSendPasswordResetEmail
    - This test calls the sendPasswordResetEmail method and checks that it returns the string "redirect:/login", which would redirect the user to the login page.
    - Also verifies that findByEmail was called once on the UserRepository with the user's email.

Iteration: 3
Story Points: 3

Favorites - Add Property to Favorites

Actors: Sarah is a user looking for suitable properties to move into with her children.

Preconditions:

- She is browsing the website for potential homes, and she clicks “Add to Favorite” to bookmark some properties to her favorites list for easy access and comparison later.

Actions:

- If she is already a registered user and signed in, the bookmark button next to “Add to Favorites” will have the color yellow, and the property is added to the joined database that stores the user's unique favorite list. She can view her favorite properties displayed in the favorite page.
- If Sarah is not signed in, she is redirected to the login page, where she has the option to either login (if she already has an account) or register for a new account.

Acceptance Criterias:

- If the user is signed in:
  - The bookmark button should have the color yellow after the user clicked on it.
  - The bookmarked property will be saved in the joined database of the user with its correct information.
  - The properties the user has bookmarked on the listings page are displayed on their favorites page.
- If the user is not signed in:
  - The user will be redirected to the sign in page.
  - The attempted bookmarked property will not be added into the user’s joined database.
- Each users’ favorite list should be independent from each other.

Tests:

- ControllersTests.java > testAddToFavorites
  - The user can add property listings to their favorites.

Iteration: 2

Story Points: 2

Favorites - Remove Property from Favorites

Actors: Peter is a signed in user looking through his favorite properties.

Preconditions:

- He clicks on the highlighted “Add to Favorites” button in the property modal.

Actions:

- The bookmark button next to “Add to Favorites” will no longer have the color yellow.
- The property is removed from the favorites page for the user.

Acceptance Criterias:

- The bookmark button should be gray after the user clicks on it.
- The bookmarked property will be removed from the database.
- The favorite list will be unique between registered users.
- The properties the user had favorited on the listings page will now no longer be displayed on the favorites page.

Tests:

- ControllersTests.java > testRemoveFromFavorites
  - The user can remove property listings to their favorites.

Iteration: 2
Story Points: 1

Favorites - View Favorites

Actors: Sarah is a registered user who has a few properties in her favorite list. She wants to see the list now.

Preconditions:

- Sarah is registered and has a user account.
- Sarah has some properties already added on her favorite list.
- Sarah is able to navigate to the favorites page.

Actions:

- When Sarah visits the favorites page, if she is logged in, a list of properties will be displayed to her, based on her list of favorites.
- If she is not logged in, she will be redirected to the login page and need to login.

Acceptance Criterias:

- If the user is signed in:
  - The favorites page should be able to be accessed.
  - The user should be able to see a list of their favorites.
  - The user should be able to see details of a favorite entry by clicking on it.
- If the user is not signed in:
  - The favorites page should not be accessible.
  - The user will be redirected to the sign in page.
- Each users' favorite list should be independent from each other.

Tests:

- ControllersTests.java > testGetFavorites
  - The users list of favorites can be obtained.

Iteration: 2
Story Points: 1

Admin - Login and Logout

Actors: John, an admin user of the site, wants to login to the site as an admin. After he is done with everything he will then logout.

Preconditions:

- From the homepage, John clicks the "Sign In" button on the top right. Then, he clicks the "Admin? Login Here" button, which redirects him to the admin login page.
- After filling in the form, he clicks the "Sign In" button.
- After completing what he needs to do, John then clicks the "Admin Logout" button to logout.

Actions:

- Validate the form to check if all fields are filled by searching the admin database to find the admin account by email and password entered.
- If an admin account is found:
  - Store login information in John's session cookie, and John is redirected to a protected admin page.

- If the credentials do not match any in the database:
    - John will remain on the admin sign in page.
- When John clicks the “Admin Logout” button to logout, the session will be destroyed and he will be redirected to the home page.

Acceptance Criterias:

- The admin user needs to be able to visit the registration page.
- The information the admin user has entered needs to be validated.
- If invalid credentials are entered:
    - The admin user needs to not be able to login and no session information should be stored.
    - The admin user needs to stay on the same login page.
- If the correct credentials are entered, the admin user needs to be logged in, and his browser now needs to contain the correct session information.
- When John clicks the “Admin Logout” button to logout, the session will be destroyed and he will log out.

Tests:

- AdminControllerTests.java > testShowAdminLoginPage_NotLoggedIn_ShouldReturnLoginPage
    - The admin login page can be displayed, and if the user is not logged in as admin, the same login page should be displayed.
- AdminControllerTests.java > testShowAdminLoginPage_LoggedIn_ShouldRedirectToAdminHomepage
    - If the admin user is logged in, the protected admin page can be displayed instead.
- AdminControllerTests.java > testAdminLogin_WithValidCredentials_ShouldRedirectToAdminHomepage
    - When the admin user logs in, if the credentials are valid, they should be redirected to the homepage.
- AdminControllerTests.java > testAdminLogin_WithInvalidCredentials_ShouldReturnToLogin
    - When the admin user logs in, if the credentials are invalid, they should be still on the login page.
- AdminControllerTests.java > testAddAdmin_ShouldCreateAdminAndRedirect
    - It is possible for the add admin endpoint to create admin.

Iteration: 2

Story Points: 2

Admin - View User List

Actors: Aman is the admin, and he wants to view how many customers are Realtorest users.

Preconditions:

- Aman is signed in as an admin.

Actions:

- After signing in with admin credentials and clicking on either the ‘View Users’ tab on the navigation bar or the ‘View Users’ button at the center of the admin home page, Aman will be navigated to the users viewing page.
- Aman can see how many Realtorest users with their basic information: first and last name, email, and mailing list subscription (as boolean value).

- Aman also wants to send the Realtorest users, who subscribed to the mailing list, a greeting email, so he enters the email subject and content and clicks the 'Send Mail' button.

Acceptance Criterias:

- The user signed in with admin credentials from the admin sign in page.

Iteration: 3

Story Points: 1

Admin - View Property List and Create New Property Entry

Actors: Aman is an admin, and he wants to view all of the properties listed on Realtorest.

Preconditions:

- Aman is signed in as an admin.

Actions:

- After clicking on either the 'Manage Listing' tab on the navigation bar or the 'Edit Listings' button, Aman will be navigated to the view property page, where he will be presented with a table contains all listings' information: owner's name, location, price, area, number of bedrooms and bathrooms, featured boolean value, description, and a set of three actions button where he has the option to add new images to the property, delete the property, or edit the property.
- If Aman decides to add a new property to the listing, he can fill out the form below the listing table. After clicking on the 'Add Property' button to submit the form, the new property will be added to the listing database.

Acceptance Criterias:

- Aman is signed in using admin credentials, and he signs in on the admin sign-in page.
- Upon clicking on the 'Add Property' button, all of the input fields of the form need to be filled with the correct format.

Test:

- PropertyControllerTest.java > testEditListings_Admin
  - Admin access to edit listing, should show edit page.
- PropertyControllerTest.java > testAddProperty_ShouldSavePropertyAndRedirect
  - New properties should be able to be added into the database.

Iteration: 3

Story Points: 2

Admin - Edit Details of Property Entry

Actors: John is an admin, and he wants to modify some information of an existing property.

Preconditions:

- John must be logged in as an admin and on the 'Manage Listings' page.
- From the 'Manage Listings' page, John can view a list of all properties, which have an 'Edit' button under the actions tab. When John clicks this, he will be redirected to the 'Edit Property' page, which shows him a form of editing property's information where he can change the details of the property.

Actions:

- Once John has filled out the form with valid inputs, he can click on the 'Update Property Information' button.

- After clicking the ‘Update Property Information’ button, the database will save the changes to the property and John will be redirected to the ‘Manage Listings’ page.

Acceptance Criterias:

- The admin user needs to be signed in to edit a property listing. If not, they will be redirected to the admin login page.
- The edit property form will need to be filled out before the admin can press the ‘Update Property Information’ button.
- Once redirected back to the ‘Manage Listings’ page, the property that was edited should display the updated property information.

Iteration: 3

Story Points: 1

Admin - Edit Images of Property Entry

Actors: John is an admin, and he wants to update the images of a property.

Preconditions:

- John must be logged in as an admin and on the ‘Edit Image’ page, this can be accomplished after he clicked on a property’s ‘Image’ action button on the ‘Manage Property’ page.
- From the ‘Edit Image’ page, John can see a list of images associated with the property.

Actions:

- If John wants to delete an existing image, he can click on the ‘Delete’ button, and the image will be deleted from the database and no longer associated with the property.
- If John wants to add a new image of the property, he will need to paste the link as an image address to the form below the image table and click the ‘Add Image’ button.

Acceptance Criterias:

- The admin user needs to be signed in to edit a property listing. If not, they will be redirected to the admin login page.
- The image that is deleted needs not to be in the database anymore.
- After clicking the ‘Add Image’ button and assuming the image’s address is valid, the image must be stored in the database and displayed on the property listing.

Iteration: 3

Story Points: 1

Admin - Delete Property Entry

Actors: John is an admin, and he wants to delete a listing since the property has been sold.

Preconditions:

- John must be logged in as an admin and on the ‘Manage Listings’ page.
- From the ‘Manage Listings’ page, John can view a list of all properties, which have an ‘Delete’ button under the actions tab.

Actions:

- After clicking the ‘Delete’ button, the property will be deleted from the system with all of its associated images.

Acceptance Criterias:

- The admin user needs to be signed in to edit a property listing. If not, they will be redirected to the admin login page.

- The property's information will not be stored in the database anymore, in addition with its associated images.

Test:

- PropertyControllerTest.java > testDeleteProperty
    - Deleting a property should be possible.
    - Should be redirected to the correct page after deletion.

Iteration: 3
Story Points: 1

Admin - Send Mail to Mailing List

Actors: Maya is an admin, and she wants to send an email to her Realtorest user.
Preconditions:

- The intended recipients must be enrolled in the mailing list.
- Maya is at the 'View Users' page, where she can send the email.

Actions:

- After entering the email subject, Maya can enter the email content that she wants to send to her Realtorest customer.
- By clicking the 'Send Mail' button, the email will be sent to all of her Realtorest users, who are enrolled in the mailing list.

Acceptance Criterias:

- The admin user needs to be signed in to edit a property listing. If not, they will be redirected to the admin login page.
- The recipient will receive the email sent by Maya.

Iteration: 3
Story Points: 3

Mortgage Calculator

Actor: Jonathan is on the Mortgage Calculator page, and wants to find out what the mortgage payments may look like for a property
Preconditions:

- Jonathan has entered the price of the property and a downpayment for the property that does not exceed the property price.
- Jonathan has chosen an amortization period, payment frequency and manually entered an interest rate between 0 and 50%.

Actions:

- Jonathan clicks the 'Calculate your Mortgage" button, which checks if he has entered valid inputs into the downpayment and property price field, calculates and displays his mortgage price next to the 'Your Mortgage' text.
- After clicking the 'Calculate your Mortgage', Jonathan is alerted to fill out an interest rate between 0 and 50% if he hasn't already entered a rate.
- Jonathan clicks the 'Calculate your Payment" button, which checks if he has entered a valid interest rate, calculates and displays his payment details to the 'Your Payment Details' text.

Acceptance Criterias:

- Jonathan must fill out the property price and downpayment field with valid numbers, and the down payment must be less than the property price before he can successfully calculate his mortgage.
- Jonathan must have already calculated his mortgage, and must enter a valid interest rate percentage from 0 to 50% before he can calculate his payment details.

Iteration: 3
Story Points: 1

Pop-Up Form

Actors: Mary is on the Listings page and is looking through the properties that are available.
Preconditions:

- Mary has been on the Listings page for two continuous minutes, looking through the properties.

Actions:

- A popup appears asking Mary if she needs assistance and asks her to fill in her name, email, and phone number.
    - Mary can choose to close the popup by clicking the “x” on the top right of the form or,
    - Mary can choose to submit her information so that the realtor can reach back to her for further assistance.

Acceptance Criterias:

- Mary fills out her information in the popup and clicks the submit button.
- That information is sent in an email to the realtor’s mail once submitted.

Iteration: 3
Story Points: 1

Velocity Report

Iteration 1

In iteration 1, we have completed setting up the website, the property listings page, and the functionality to register, login, and logout for regular users. This adds up to 7 story points in total, which is equal to 3.5 points per week.

Iteration 2

In iteration 2, we have completed displaying the detailed view of property listings, the functionality to filter properties by attributes, the admin user system, the favorites system. This adds up to 10 story points in total, which is equal to 5 points per week.

We have also implemented the unit test system, and the properties database to store property information.

Generally speaking, we are getting more things done in iteration 2 compared to the previous iteration. The difference in amount of work done is likely even bigger than the story points calculation entails. Since some work is improving the features done in iteration 1, like adding a check to password security, or actually implementing the table to store property data in our database. We also worked on getting unit tests implemented for both features in iteration 1 and 2.

Iteration 3

In iteration 3, we have implemented the Mailgun API for sending out emails to mailing lists and resetting passwords, and the Google Maps Embed API for showing the map around each property listing. We have also added the feature for users to change their information or password and sign up for the mailing list, use the mortgage calculator, and fill out a pop up form to receive help. More functionalities have also been added to the admin side, including viewing a list of all users, sending out mails, and viewing a list of all properties to add, edit, or delete properties. This adds up to 18 story points in total, which is equal to 9 points per week.

We have also improved a lot of old features, like storing the image in a database instead of hard coded in, checking if the user's email or username have been taken already when registering, automatically logging in the user when registering, etc.

We have also fixed many bugs, including caching causing protected pages can still be seen by clicking "back", filter bar issues, property listings page having a different URL before and after applying filters, visiting the main homepage when you are still logged in as an admin will cause the page to not be able to load, etc.

There are also a lot of refactoring being done, like modularization by extracting components out from the only controller, organizing URL for the pages, using Thymeleaf fragments to extract the navigation bar from every page, writing a lot of tests for features with missing tests previously, etc.

In conclusion, iteration 3 is the iteration where we got the most amount of work done. Not only were we able to implement a lot of new features, the code quality has actually improved too. The amount of technical debt, or we can say "code rot", have been greatly reduced by the heavy amount of refactoring being done (examples includes pull request #158, #173, #230, etc), and the implementation of more coded tests that can be run before merging pull requests.

Dividing Work

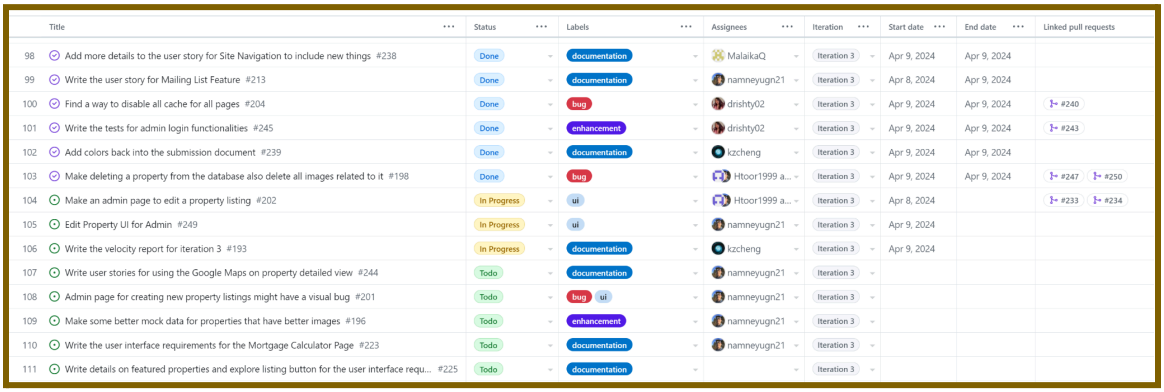
In iteration 1, a problem we had was dividing the work into "frontend" and "backend". This turned out to be causing a lot of troubles, since the person who worked on the "backend" worked on creating endpoints for GET and POST requests, but there isn't any UI to actually test if those endpoints work.

Meanwhile, the person working on "frontend" needed those endpoints to actually read and write data from the database, and without them, the UI is just talking to imaginary endpoints, and cannot be tested at all. As a result, the last few days of iteration 1 is very heavily loaded, because a large amount of time is being spent on connecting the frontend with the backend, which often means doing pretty big rewrites of parts that need the UI to interact with the database.

In iteration 2, we have shifted towards assigning work to each person that is more "feature" based. When allowing each person to work on a new "feature" that includes both part of UI and database interactions, work being done can be tested by the person working on it to make sure it works, before

being merged into the main branch. This way, the main branch always has a copy of the website that is working.

In iteration 3, we've tried to learn from Agile methodologies and use a workflow where, instead of assigning work to each person, make a list of things we need to do, and have each person check out what they want to work on now and just work on it. We've been using GitHub Issue posts to list things we need to do, and GitHub Projects to manage the list. As shown below, this allows us to see who wants to work on what, because each person can self assign Issues to themselves. This makes sure that no one will be doing duplicated work, and everyone can adjust their amount of workload based on their schedule.



	Title	Status	Labels	Assignees	Iteration	Start date	End date	Linked pull requests
98	Add more details to the user story for Site Navigation to include new things #238	Done	documentation	MalaikaQ	Iteration 3	Apr 9, 2024	Apr 9, 2024	
99	Write the user story for Mailing List Feature #213	Done	documentation	namneyugn21	Iteration 3	Apr 8, 2024	Apr 9, 2024	
100	Find a way to disable all cache for all pages #204	Done	bug	drishty02	Iteration 3	Apr 9, 2024	Apr 9, 2024	#240
101	Write the tests for admin login functionalities #245	Done	enhancement	drishty02	Iteration 3	Apr 9, 2024	Apr 9, 2024	#243
102	Add colors back into the submission document #239	Done	documentation	kzcheng	Iteration 3	Apr 9, 2024	Apr 9, 2024	
103	Make deleting a property from the database also delete all images related to it #198	Done	bug	Htoor1999 a...	Iteration 3	Apr 9, 2024	Apr 9, 2024	#247 #250
104	Make an admin page to edit a property listing #202	In Progress	ui	Htoor1999 a...	Iteration 3	Apr 8, 2024		#233 #234
105	Edit Property UI for Admin #249	In Progress	ui	namneyugn21	Iteration 3	Apr 9, 2024		
106	Write the velocity report for iteration 3 #193	In Progress	documentation	kzcheng	Iteration 3	Apr 9, 2024		
107	Write user stories for using the Google Maps on property detailed view #244	Todo	documentation	namneyugn21	Iteration 3			
108	Admin page for creating new property listings might have a visual bug #201	Todo	bug ui	namneyugn21	Iteration 3			
109	Make some better mock data for properties that have better images #196	Todo	enhancement	namneyugn21	Iteration 3			
110	Write the user interface requirements for the Mortgage Calculator Page #223	Todo	documentation	namneyugn21	Iteration 3			
111	Write details on featured properties and explore listing button for the user interface requ... #225	Todo	documentation		Iteration 3			

The plan seems to have worked, and we were able to get much more work in iteration 3. There weren't really any problems where people worked on duplicated things, and work was able to be spread out pretty evenly across the iteration.

User Interface Requirements

Home Page

</>

As described in the user stories, the first page the user sees when visiting the site should be the home page. The nav bar on top of the screen has some buttons on the right that can be clicked, including the information of the website owner, the link to jump to property listing view, and login or logout button.

Users can scroll down to see a list of featured properties. Users can see the property's image, price, area, number of bathrooms and bedrooms, and location. And, after scrolling to the very bottom of the home page, the user can click on the 'Explore Our Listing' button, where it will navigate users to the property listing page that lists all of the properties.

Navigation Bar

For the user page, on the top left of the nav bar, the website logo is displayed. This is also a button that you can click to go back to the home page. The top right corner displays the login button, but if the user is already logged in, it will display the logout button instead.

When clicking the “About Us” button on the nav bar, a detailed view will drop down from the top, while dimming everything else on the website. The view will contain information about the client, who is the website owner. This includes the client’s photo, their contact information, a short self introduction, and links to their social media pages.

On the top right, there are also the “Listing”, “Mortgage Calculator”, and “Favorites” buttons, which lead the user to the corresponding pages.

Property Listing Page

</properties>

After clicking on “Listing” on the navigation bar, the user will be directed to the property listing page that will display different properties with information such as price, location, and number of baths and bedrooms.

The pagination, located at the bottom of the property listing, allows users to navigate through different pages of properties. This way, the user will not have to scroll forever; instead, the user can easily jump to different sections and find what they are looking for faster.

The property listing page has a filter and sorting bar on top, letting users narrow down their search for an ideal property. The bar prompts users to refine their search through options like specifying a city, sorting properties by price, and selecting the desired number of bedrooms and bathrooms. There's also a spot to search for a house by its name. Once the user picks what they want, just hit the "Apply Filters" button to the filtered results.

Clicking on a property entry will show the details of a property. The screen will darken and a popup card will appear, showing the images, information, detailed description, a Google Maps view of the location, and contact information for Aman at the bottom. There is also a bookmark icon, and clicking on it highlights it and adds the property to the user’s favorites. Unhighlighting the bookmark removes the property from the user’s favorites. The user can view all the properties they have favorited through clicking “Favorites” on the navigation bar.

Account - Login Page

</login>

Visiting the login page, the user can see a form where they can enter the email and password associated with their account to login. There is also a link prompting the user to register an account if they do not have one. Clicking it will redirect them to the register page.

Below the login form, there is a button for leading the user to the admin login portal. There is also a “Forgot Password” button that leads the user to the corresponding page.

Account - Register Page

</register>

On the register page, there is a form, which includes the username, email, password, and a password confirmation slot. After entering those details, the user can click the register button to create the account. After so, the user will be redirected back to the home page.

Account - Forgot Password Page

</forgotpassword>

On the forgot password page, there is a form, which has two input fields, email address and confirm email address. After the user who has forgotten their password enters their email and the input on both the fields match, a mail is sent to the user's email with a link that leads to a page to reset the user's password. If the emails do not match, an alert pops up saying “Emails do not match!”. If the email entered does not have an account, the user is redirected to the register page.

Account - Password Reset Page

</resetpassword>

On the reset password page, there is a form, which has two input fields, new password and confirm new password. This page is reached through the link sent to the user’s email after they have entered their email and clicked submit on the forgot password page. Once the user has put in the new password and the input on the confirm new password matches, clicking the submit button would lead to the page being redirected to the login where the user can now sign in to their account with the new password. If the passwords do not match, an alert pops up saying “Passwords do not match!”.

Account - User Settings Page

</settings>

User settings page presents the user with a form that is filled with the current user’s information: first and last name, username, email address, password, and checkbox of mailing subscription. By clicking on a field, the user can modify their information, or by toggle the checkbox, the user can choose to opt-in or out of the mailing list. Once a field is modified, the ‘Save’ button will be enabled, and the user is required to click on the button in order to save their new information. The user can also navigate back to the home page by clicking on the ‘Back’ button.

Favorites Page

</favorites>

Properties that the user has favorited are displayed on the page if they are logged in. A user who is not logged in cannot access the favorites page and is redirected to the login page. If a user wants to remove a property from their favorites they have to unhighlight the bookmark on the listing and the property is then removed from the user’s favorites page.

Admin - Home Page

</admin>

Admin page is designed by keeping the security in mind, it requires admin login with email and password to access the protected admin page. Once authenticated, the admin is directed to a protected admin area, ensuring that sensitive operations are kept secure.

Upon landing on the admin home page, admin can navigate to the listing management page by clicking on the 'Manage Listing' tab on the navigation bar or clicking on the 'Edit Listings' button in the center of the page. In addition, admin can also go to the users page that will display a list of all *Realtorest* users and allow admin to send emails to the subscribed users, by clicking on the 'View Users' tab located in the navigation bar or the button below the 'Manage Listing' button. Finally, admin can also sign out and end their session by clicking on the 'Sign Out' tab on the navigation bar.

Admin - Navigation Bar

For the admin pages, the navigation bar includes a logout option located at the top-right corner, allowing the admin to securely exit their session. After logging out, it takes the admin to the home page.

On the top right, there are also the "Manage Listing" and "View Users" buttons, which leads to the corresponding pages.

Admin - Login Page

</admin/login>

Admin will need an admin email and password that is provided by the developer in order to sign in into the admin page. In other words, using the user's credential will not allow the admin to sign into the admin page. Regular users who are at the admin login page wanting to navigate back to the user login page can click the 'Login Here' link below the 'Login' button, where they will be led back to the Realtorest customer login page.

Admin - Manage Property Listings Page

</admin/properties>

</admin/images>

In the manage listings page, admin has the ability to add new property by filling out a form with property's information: owner's name, location, price, area, detailed descriptions, number of bedrooms and bathrooms, and if the property will be featured or not.

The page also displays a table containing every property that is listed on the website. The table will display the property's information: owner's name, location, price, area, number of bedrooms and bathrooms, featured information (as boolean value), description. In addition, the table also displays a set of three action buttons: 'Images' that allows admin to navigate to the image page and add new image, 'Delete' that allows admin to delete the listing property, and 'Edit' that allows admin to navigate to the edit property page where they can modify the information of the property.

In editing the property page, admin will be presented with a form filled with the property's current information, and they can choose which fill to modify. By clicking the 'Save' button, the form will be submitted and the property's information will be changed in the database.

In the adding image page, admin will be presented with a table that contains the property's ID and all its associated images' ID and addresses (of where the images are stored). Admin will be presented with a form where they need to add the link of the image's address, and after clicking the 'Add Image' button, the image's address will be stored in the database.

Admin - User List and Mailing List Page

</admin/users>

Users and Mailing page is designed for admin keeping track of the number of registered users and allow admin to send emails to users who are subscribing to the mailing list.

The page will display a table containing the users' basic information: first and last name, email, subscription to the mailing list (displayed as boolean value). Under the users table, admin can write and send email to users who are subscribing to the mailing list, by entering the email subject and mailing content.

Mortgage Calculator Page

</mortgage>

On the mortgage calculator page, users are required to enter the property value, down payment made, interest rate from 0.0% to 50.0%, and they need to choose the amortization period varies from 5 years to 30 years (with the options that are 5 years apart) and how frequent they will make a payment: monthly or bi-weekly.

After clicking the 'Calculate your Mortgage' and assuming that the values the user entered are valid, the page will display the principal mortgage amount, that is the property value subtracted from the down payment made, and the calculated payment the user needs to make monthly or bi-weekly.

Contact Info Popup

The popup appears after the user has been on the Listings page for two minutes asking for them to fill out their information for help. It asks them for their name, email, and phone number. Once the user clicks 'Submit', the information entered in the fields of the popup is sent to the realtor's email. The realtor can then reach back to the user to assist them personally.

User Interface Mockups

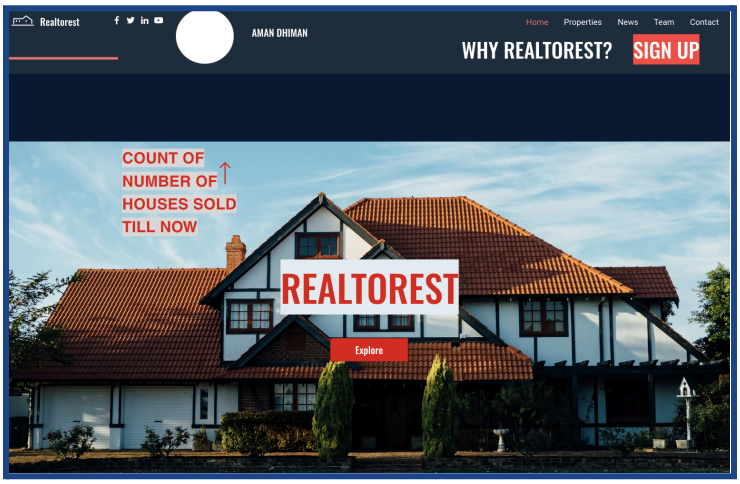


Figure 1. Home Page

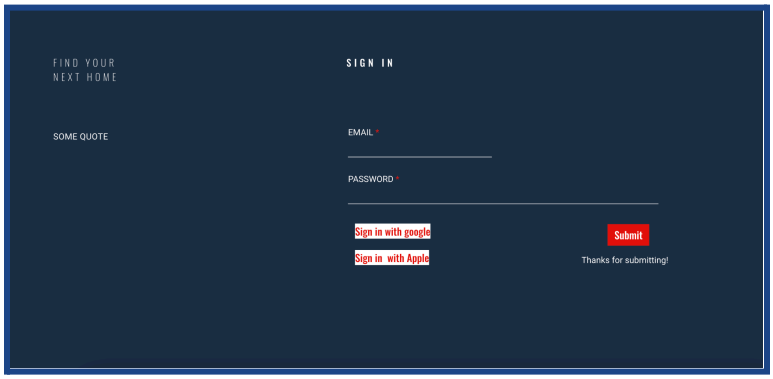


Figure 2. Login Page

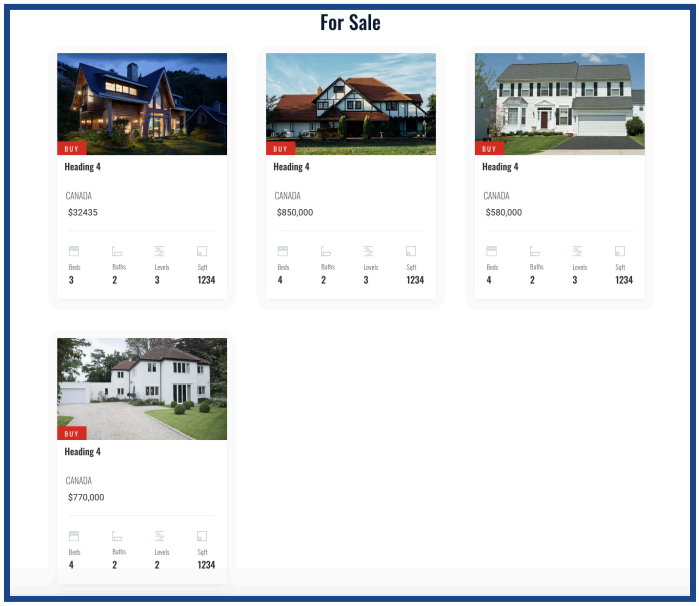


Figure 3. Properties Listing View