

# **CPSC 471 Project Final Report**

## **Real Estate Database System Group 64**

**Dora Tan - 30045844 - Tutorial T06**  
**Sydney Kwok - 30073206 - Tutorial T07**  
**Shavonne Tran - 30041912 - Tutorial T07**

## **Abstract**

This report will provide an introduction to the problem the system was designed to address and what type of system was created to solve it. It will summarize the key features of the system and discuss the targeted users. There will be a project design section to showcase a detailed ER diagram and relational model of the completed system. Through this report, readers will have a clear picture of the functionality offered by the API used in the system. The report will also describe details about the DBMS used for the system, API documentation, and include an attached user manual which explains all of the system features and functionalities to new users.

## **Introduction: The Problem & Our Solution**

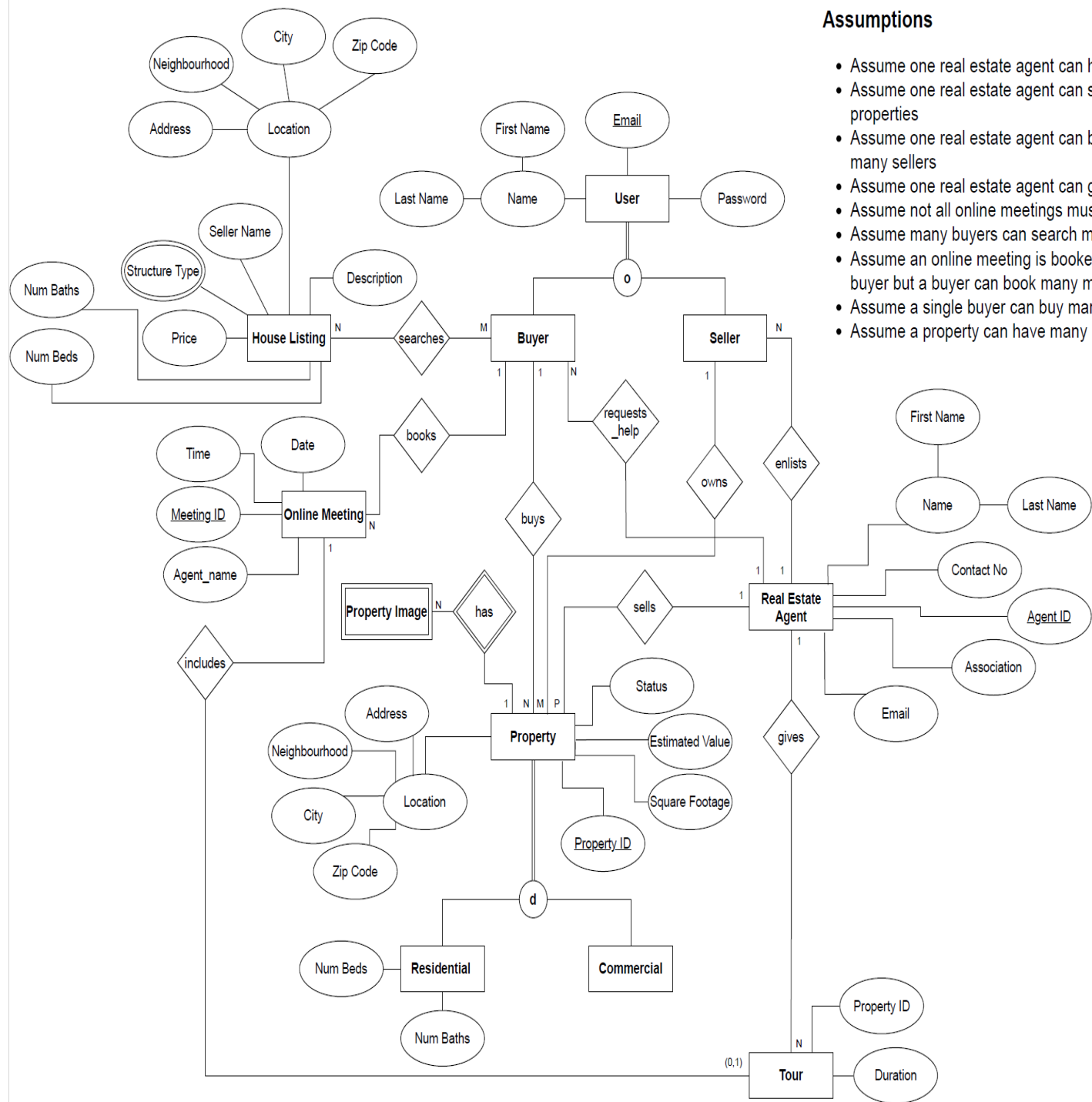
Searching for a new property that fits your standards can be difficult for many people. There is a lot to consider when trying to find a house that accommodates all your needs within your ideal budget range. Our system addresses this problem through a simple interactive website which caters to the user's wants and desires to find the best homes. We have also created a database to store related real estate information according to user selected criteria. Users indicate their preferred search parameters including the property type, city, zip code, price range, number of bedrooms, and number of baths. They can then browse and view details of homes that satisfy the user specified criteria and select houses that they are interested in visiting with a realtor. The user can even schedule a meeting with an agent through the system to get a tour of the property and at this time, the realtor will address any questions and concerns the user may have.

## **Project Design Section**

Our system has three different types of users, namely, a buyer, seller, and real estate agent.

- **Buyers** have the ability to: search for properties by specifying a number of search parameters, view property details, save properties to their bookmarks, book meetings and tours with real estate agents, and view or delete their scheduled meetings.
- **Sellers** have the ability to: search for properties, list their own properties for sale, and view or delete their own listings.
- **Agents** have the ability to: search for properties, view property details, view details for or cancel their scheduled meetings with buyers, mark properties as sold, and upload images to property listings.
- All user types also have login and registration processes.

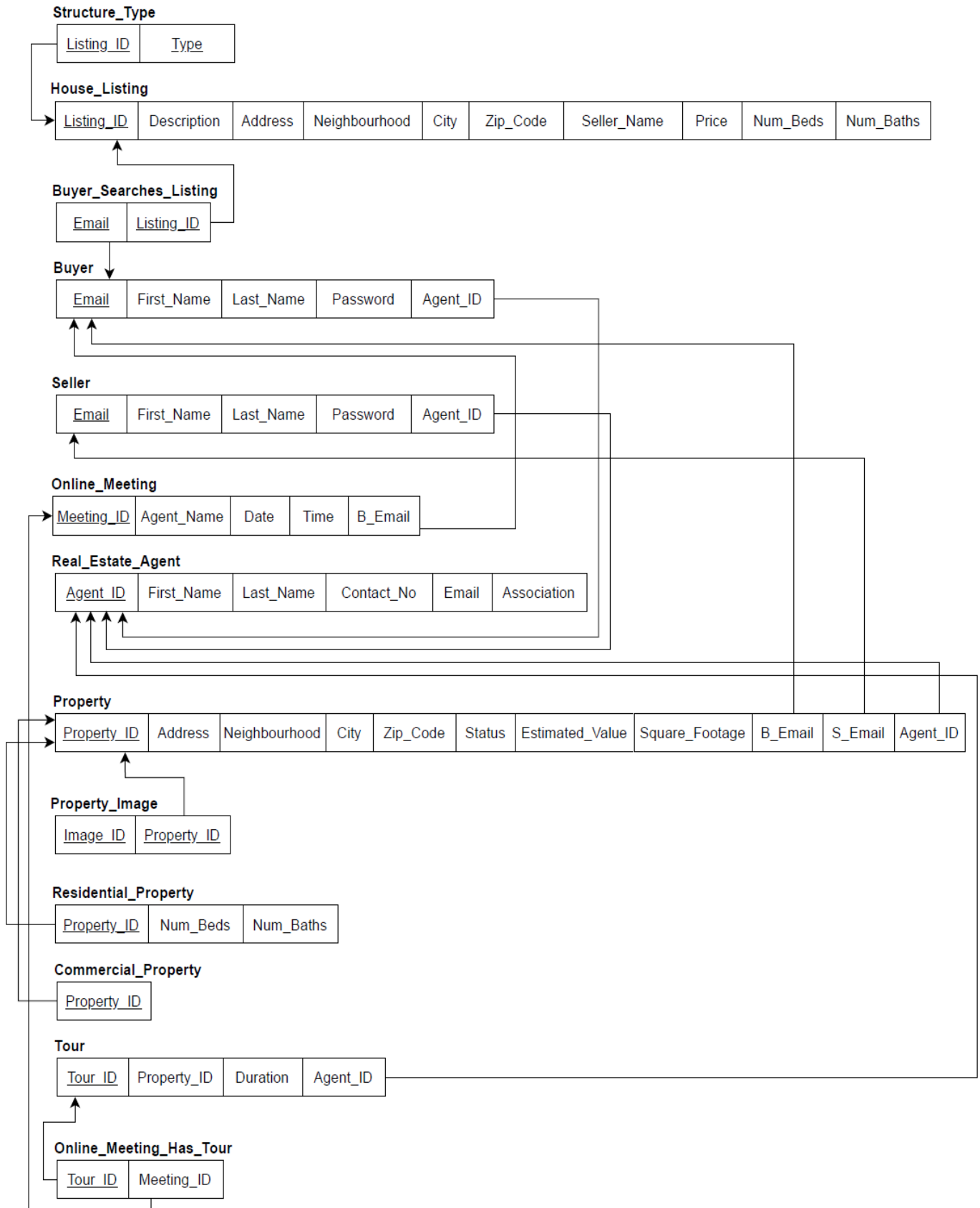
We have attached a complete ER diagram of our real estate system below.



## Assumptions

- Assume one real estate agent can help many buyers
- Assume one real estate agent can sell many properties
- Assume one real estate agent can be enlisted by many sellers
- Assume one real estate agent can give many tours
- Assume not all online meetings must include a tour
- Assume many buyers can search many houses
- Assume an online meeting is booked by only one buyer but a buyer can book many meetings
- Assume a single buyer can buy many properties
- Assume a property can have many images

## Implementation Section



Pictured above is a complete relational model diagram of the system on the following page. This model was obtained from our ER diagram by following the algorithm for converting an entity relationship diagram to a relational schema diagram. There were no significant or unusual decisions made during this process. We only made one minor decision to create a primary key for the Tour and House\_Listing tables because our ER diagram did not provide them with one.

The DBMS we selected for the implementation of the project was MySQL. Below is the set of SQL statements for each of the transactions implemented. Note that, in order to prevent SQL injections, along with minimizing the amount of user-typed input, we also made use of prepared statements. So, for all queries that involved user-provided data, we used prepared statements so that we could write the SQL command and the user-provided data separately. In this way, our SQL commands are executed safely, and in a way that prevents SQL injection vulnerabilities.

**Login:** (almost identical for seller and agent)

```
// check if user is buyer
$query = "(select Password from buyer where Email = ?) limit 1";
// create prepared statement
$stmt = mysqli_stmt_init($conn);
if (!mysqli_stmt_prepare($stmt, $query)) {
    echo "Failed to prepare statement";
} else {
    // bind parameters, "s" for type string
    mysqli_stmt_bind_param($stmt, "s", $email);
    // execute the query
    mysqli_stmt_execute($stmt);
    // get the result
    $result = mysqli_stmt_get_result($stmt);

    // check if result is good & we have atleast one result
    if ($result && mysqli_num_rows($result) > 0) {
        // get user data
        $user_data = mysqli_fetch_assoc($result);
        // check if user data's password (from db) matches user supplied password
        if ($user_data['Password'] == $password) {
            // if true, assign session id
            $_SESSION['Email'] = $email;
            // and account type
            $_SESSION['Account_Type'] = "buyer";
        }
    }
}
```

## Registration:

- Buyer & Seller

```
if ($account_type == "b") {  
    $query = "insert into buyer (Email,First_Name,Last_Name>Password) values (?, ?, ?, ?)";  
} else {  
    $query = "insert into seller (Email,First_Name,Last_Name>Password) values (?, ?, ?, ?)";  
}  
$stmt = mysqli_prepare($conn, $query);  
mysqli_stmt_bind_param($stmt, "ssss", $email, $first_name, $last_name, $password);  
mysqli_stmt_execute($stmt);
```

- Agent

```
$query = "insert into real_estate_agent (First_Name,Last_Name>Contact_No>Email>Password>Association)  
values (?, ?, ?, ?, ?, ?)";  
$stmt = mysqli_prepare($conn, $query);  
mysqli_stmt_bind_param($stmt, "ssssss", $first_name, $last_name, $contact_no, $email, $password, $association);  
mysqli_stmt_execute($stmt);
```

## Search Properties:

For property searching, we had 16 cases for each of commercial & residential property, coming out to a total of 32 cases.

1. User chooses residential/commercial, sets city, sets max price, sets # of bedrooms, sets # of bathrooms
2. User chooses residential/commercial, sets city, sets max price, sets # of bedrooms, does not set # of bathrooms
3. User chooses residential/commercial, sets city, sets max price, does not set # of bedrooms, sets # of bathrooms
4. User chooses residential/commercial, sets city, sets max price, does not set # of bedrooms, does not set # of bathrooms
5. User chooses residential/commercial, sets city, does not set max price, sets # of bedrooms, sets # of bathrooms
6. User chooses residential/commercial, sets city, does not set max price, sets # of bedrooms, does not set # of bathrooms
7. User chooses residential/commercial, sets city, does not set max price, does not set # of bedrooms, sets # of bathrooms
8. User chooses residential/commercial, sets city, does not set max price, does not set # of bedrooms, does not set # of bathrooms
9. User chooses residential/commercial, does not set city, sets max price, sets # of bedrooms, sets # of bathrooms
10. User chooses residential/commercial, does not set city, sets max price, sets # of bedrooms, does not set # of bathrooms
11. User chooses residential/commercial, does not set city, sets max price, does not set # of bedrooms, sets # of bathrooms

12. User chooses residential/commercial, does not set city, does not set max price, sets # of bedrooms, sets # of bathrooms
13. User chooses residential/commercial, does not set city, sets max price, does not set # of bedrooms, does not set # of bathrooms
14. User chooses residential/commercial, does not set city, does not set max price, does not set # of bedrooms, sets # of bathrooms
15. User chooses residential/commercial, does not set city, does not set max price, sets # of bedrooms, does not set # of bathrooms
16. User chooses residential/commercial, does not set city, does not set max price, does not set # of bedrooms, does not set # of bathrooms

In order to simplify the process of covering all of these cases, we made use of a number of flags. Since there are so many cases here, we will show the SQL statements for just one case, but if you'd like to see more, we encourage you to view the searchProperty.php file in our source code.

```
$query = "(SELECT * FROM `residential_property` AS r NATURAL JOIN `property` AS p
WHERE p.City = ? and p.Estimated_Value >= '$min_price' and
p.Estimated_Value <= '$max_price' and p.Num_Beds = '$beds' and p.Num_Baths = '$baths')";
```

### View Property Details & Property Images:

```
// get property info
$query = "SELECT * FROM property WHERE Property_ID = '$prop_id' limit 1";
$result = mysqli_query($conn, $query);
```

```
$img_query = "(SELECT * FROM `property_image` WHERE Property_ID = '$prop_id')";
$images = mysqli_query($conn, $img_query);
```

### Bookmark Property:

- Bookmark Property

```
$email = $user_data['Email'];

$query = "insert into buyer_bookmarks_property (Email,Property_ID) values ('$email', '$prop_id')";
mysqli_query($conn, $query);
```

- Retrieve Bookmarks of Particular Buyer Account

```
// get the bookmarks of the user
$query = "select * from buyer_bookmarks_property where Email = '$email'";
$result = mysqli_query($conn, $query);
```

- Delete Bookmark

```
// delete the corresponding email-property tuple from the buyer_bookmarks_property relation
$query = "delete from `buyer_bookmarks_property` where `buyer_bookmarks_property`.`Email` = '$email'
and `buyer_bookmarks_property`.`Property_ID` = '$prop_id'";
mysqli_query($conn, $query);
```

## Book Meeting/Tour:

```
// add the meeting to the online_meeting relation
$query = "insert into online_meeting (Agent_ID, Date, Time, B_Email, Message)
        values ('$agent', '$date', '$time', '$b_email', ?)";
$stmt = mysqli_prepare($conn, $query);
mysqli_stmt_bind_param($stmt, "s", $message);
mysqli_stmt_execute($stmt);
// get the meeting id of the meeting that was just created in the db
$meeting_id = mysqli_insert_id($conn);

if ($tour == "y") {
    // if the user wants a tour, add it to the tour relation
    $query = "insert into tour (Property_ID, Agent_ID) values ('$property', '$agent')";
    mysqli_query($conn, $query);
    // get the tour id of the tour that was just created in the db
    $tour_id = mysqli_insert_id($conn);
    // & add this info to the online_meeting_has_tour relation
    $query = "insert into online_meeting_has_tour (Tour_ID, Meeting_ID)
            values ('$tour_id', '$meeting_id')";
    mysqli_query($conn, $query);
}
```

## View/Delete Meeting (Buyer & Seller):

- View Meetings

```
// depending on account type, get the meetings of the user
$email = $user_data['Email'];
if ($_SESSION['Account_Type']=="buyer") {
    $query = "select * from online_meeting where B_Email = '$email'";
} else if ($_SESSION['Account_Type']=="agent") {
    // get agent id from real_estate_agent based on their email
    $query = "select Agent_ID from real_estate_agent where Email = '$email'";
    $result = mysqli_query($conn, $query);
    if ($result && mysqli_num_rows($result)>0) {
        $agent_data = mysqli_fetch_assoc($result);
        $agent_id = $agent_data['Agent_ID'];
        $query = "select * from online_meeting where Agent_ID = '$agent_id'";
    } else {
        echo "There was an error";
        die;
    }
}
$result = mysqli_query($conn, $query);
```



```

// check if meeting has tour
$meet_id = $meet_data['Meeting_ID'];
$query = "select Tour_ID from online_meeting_has_tour where Meeting_ID = '$meet_id' limit 1";
$has_tour = mysqli_query($conn, $query);
if ($has_tour && mysqli_num_rows($has_tour) > 0) {
    // get tour data
    $tour_data = mysqli_fetch_assoc($has_tour);
    $tour_id = $tour_data['Tour_ID'];
    $query = "select * from tour where Tour_ID = '$tour_id' limit 1";
    $tour_data = mysqli_query($conn, $query);
    if ($tour_data && mysqli_num_rows($tour_data) > 0) {
        $tour_data = mysqli_fetch_assoc($tour_data);
        $property_id = $tour_data['Property_ID'];
    }
}
}

```

- Delete Meeting

```

// check if the meeting has a tour
$query = "select Tour_ID from online_meeting_has_tour where Meeting_ID = '$meeting_id'";
$result = mysqli_query($conn, $query);
if ($result && mysqli_num_rows($result) > 0) {
    // if the meeting has a tour delete from online_meeting_has_tour and tour
    $res_data = mysqli_fetch_assoc($result);
    $tour_id = $res_data['Tour_ID'];
    $query = "delete from `online_meeting_has_tour` where `online_meeting_has_tour`.`Meeting_ID` = '$meeting_id' and `online_meeting_has_tour`.`Tour_ID` = '$tour_id'";
    mysqli_query($conn, $query);
    $query = "delete from `tour` where `tour`.`Tour_ID` = '$tour_id'";
    mysqli_query($conn, $query);
}

// delete the meeting from online_meeting
$query = "delete from `online_meeting` where `online_meeting`.`Meeting_ID` = '$meeting_id'";
mysqli_query($conn, $query);

```

## List New Property For Sale:

```

// add the property details to the property relation
$query = "insert into property (Address, Neighbourhood, City, Zip_Code, Estimated_Value, Square_Footage, Num_Beds, Num_Baths, S_Email)
values (?, ?, ?, ?, ?, ?, ?, ?, '$s_email')";
$stmt = mysqli_prepare($conn, $query);
mysqli_stmt_bind_param($stmt, "ssssiiii", $address, $neighbourhood, $city, $zip, $value, $footage, $num_beds, $num_baths);
mysqli_stmt_execute($stmt);
// get the property id of the property that was just created in the db
$prop_id = mysqli_insert_id($conn);
// add the property id to commercial or residential property table based on selection
if ($prop_type=="com") {
    $query = "insert into commercial_property (Property_ID) values ('$prop_id')";
    mysqli_query($conn, $query);
} else {
    $query = "insert into residential_property (Property_ID) values ('$prop_id')";
    mysqli_query($conn, $query);
}
}

```

## View/Delete Seller's Own Listings:

- View

```
// get the listings of the user
$email = $user_data['Email'];
$query = "select * from property where S_Email = '$email'";
$result = mysqli_query($conn, $query);
```

- Delete

```
// check if the property has any tours
$query = "select Tour_ID from tour where Property_ID = '$prop_id'";
$result = mysqli_query($conn, $query);

if ($result && mysqli_num_rows($result) > 0) {
    // if the property has scheduled tours, for each tour...
    while ($tour_data = mysqli_fetch_assoc($result)) {
        // get the tour id
        $tour_id = $tour_data['Tour_ID'];

        // get the meeting_id of the meeting that the tour is included in
        $query = "select Meeting_ID from online_meeting_has_tour where Tour_ID = '$tour_id'";
        $meet_data = mysqli_query($conn, $query);
        $meet_data = mysqli_fetch_assoc($meet_data);
        $meeting_id = $meet_data['Meeting_ID'];

        // delete the corresponding tour-meeting entry from online_meeting_has_tour
        $query = "delete from `online_meeting_has_tour` where `online_meeting_has_tour`.`Tour_ID` = '$tour_id' and `online_meeting_has_tour`.`Meeting_ID` = '$meeting_id'";
        mysqli_query($conn, $query);
        // delete the tour from the tour relation
        $query = "delete from `tour` where `tour`.`Tour_ID` = '$tour_id'";
        mysqli_query($conn, $query);
        // delete the corresponding meeting from online_meeting relation
        $query = "delete from `online_meeting` where `online_meeting`.`Meeting_ID` = '$meeting_id'";
        mysqli_query($conn, $query);
    }
}
```

```
// delete the property from the residential relation (whether it exists there or not)
$query = "delete from `residential_property` where `residential_property`.`Property_ID` = '$prop_id'";
mysqli_query($conn, $query);
// delete the property from the commercial relation (whether it exists there or not)
$query = "delete from `commercial_property` where `commercial_property`.`Property_ID` = '$prop_id'";
mysqli_query($conn, $query);
// delete the property from the property image relation (even though it's not yet populated/implemented)
$query = "delete from `property_image` where `property_image`.`Property_ID` = '$prop_id'";
mysqli_query($conn, $query);
// delete the property from the property relation
$query = "delete from `property` where `property`.`Property_ID` = '$prop_id'";
mysqli_query($conn, $query);
```

## Mark Property As Sold:

```
// check if the property has any tours
$query = "select Tour_ID from tour where Property_ID = '$prop_id'";
$result = mysqli_query($conn, $query);

if ($result && mysqli_num_rows($result) > 0) {
    // if the property has scheduled tours, for each tour...
    while ($tour_data = mysqli_fetch_assoc($result)) {
        // get the tour id
        $tour_id = $tour_data['Tour_ID'];

        // get the meeting_id of the meeting that the tour is included in
        $query = "select Meeting_ID from online_meeting_has_tour where Tour_ID = '$tour_id'";
        $meet_data = mysqli_query($conn, $query);
        $meet_data = mysqli_fetch_assoc($meet_data);
        $meeting_id = $meet_data['Meeting_ID'];

        // delete the corresponding tour-meeting entry from online_meeting_has_tour
        $query = "delete from `online_meeting_has_tour` where `online_meeting_has_tour`.`Tour_ID` = '$tour_id'
        |         and `online_meeting_has_tour`.`Meeting_ID` = '$meeting_id'";
        mysqli_query($conn, $query);
        // delete the tour from the tour relation
        $query = "delete from `tour` where `tour`.`Tour_ID` = '$tour_id'";
        mysqli_query($conn, $query);
        // delete the corresponding meeting from online_meeting relation
        $query = "delete from `online_meeting` where `online_meeting`.`Meeting_ID` = '$meeting_id'";
        mysqli_query($conn, $query);
    }
}

// delete the property from the residential relation (whether it exists there or not)
$query = "delete from `residential_property` where `residential_property`.`Property_ID` = '$prop_id'";
mysqli_query($conn, $query);
// delete the property from the commercial relation (whether it exists there or not)
$query = "delete from `commercial_property` where `commercial_property`.`Property_ID` = '$prop_id'";
mysqli_query($conn, $query);
// delete the property from the property image relation (even though it's not yet populated/implemented)
$query = "delete from `property_image` where `property_image`.`Property_ID` = '$prop_id'";
mysqli_query($conn, $query);
// delete the property from the property relation
$query = "delete from `property` where `property`.`Property_ID` = '$prop_id'";
mysqli_query($conn, $query);
```

## Upload Property Image:

```
$query = "insert into property_image (Property_ID) values ('$prop_id')";
mysqli_query($conn, $query);
```

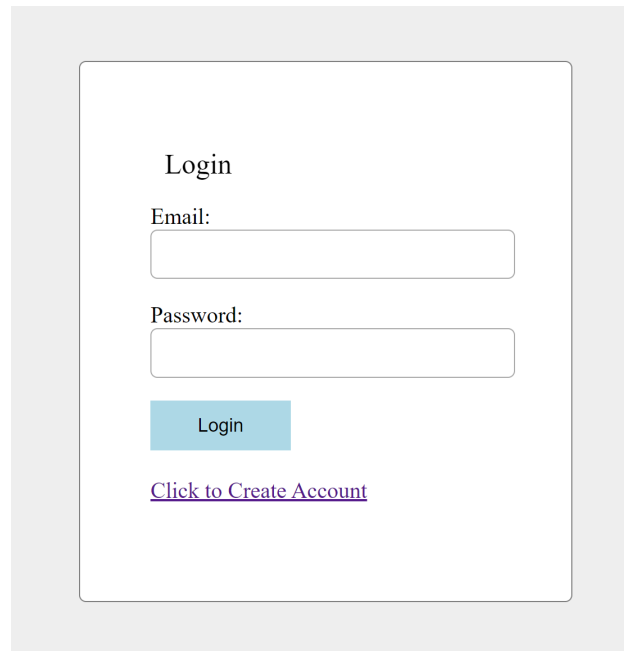
## API Documentation

We have also used the built-in API documentation feature provided by Postman to generate API documentation that describes our API endpoints and provides some sample requests and responses.

Here is a link to our API documentation using the Postman tool:  
<https://documenter.getpostman.com/view/15354571/TzJoELpX>

## **User Guide**

### **Login Page:**

A screenshot of a login page. The page has a light gray background. In the center, there is a white rectangular box with a thin gray border. Inside this box, the word "Login" is centered at the top. Below it, the label "Email:" is followed by a white input field with a thin gray border. Below the email field, the label "Password:" is followed by another white input field with a thin gray border. Below the password field, there is a blue rectangular button with the word "Login" in white text. At the bottom of the white box, there is a purple underlined link that says "Click to Create Account".

The first page users are directed to when running the system is the login page. To access the website, users must enter their login credentials that they specified during the registration process. If the user has not yet created an account, they must click the link to register for one. After entering their login credentials, the user must click the “Login” button which will direct them to the index page upon success. If the login credentials were incorrect, there will be a message informing the user to enter valid information. For testing purposes, we have pre-created three test accounts, one for each user type. If you would like to use one of these accounts, the login credentials are listed as follows, based on user type.

#### **Test Account for Buyer**

Email: testb

Password: buyer

#### **Test Account for Real Estate Agent**

Email: testa

Password: agent

### Test Account for Seller

Email: tests

Password: seller

### Registration Page:

#### User Registration

Account Type: ☐ Buyer ☐ Seller

Email:

First Name:

Last Name:

Password:

Register

[Click to Login](#)

[Click to Register as Agent](#)

#### Agent Registration

Email:

First Name:

Last Name:

Password:

Phone Number:

Association:

Register

[Click to Login](#)

[Click to Register as User](#)

To create an account for the system, users must first select an account type: buyer or seller. Then, they will be asked to enter their email, first and last names, and a password. Once they press the “Register” button, their account information will be recorded into the database and they will be redirected to the login page to sign in with the newly created credentials.

If the user is a real estate agent, they can click the link at the bottom of the page to register as an agent user. Similarly, they will be redirected to a new page and asked to enter the appropriate fields for their account. Once they press the “Register” button, their account information will be recorded into the database and they will be redirected to the login page to sign in with the newly created credentials.

## Home Page:

Logo

Logout

### Welcome to the Real Estates Home Page!

Hello, buyer testf.

Our business hours are 9am-7pm for 7 days a week. Select one of the following options to begin your property search.

[Click to Search for Properties](#)

[Click to Book a Meeting With a Real Estate Agent](#)

[Click to View The Properties You've Bookmarked](#)

[Click to View Your Meetings](#)

Logo

Logout

### Welcome to the Real Estates Home Page!

Hello, seller testf.

Our business hours are 9am-7pm for 7 days a week. Select one of the following options to begin your property search.

[Click to Search for Properties](#)

[Click to Sell a Property](#)

[Click to View Your Listings](#)

Logo

Logout

### Welcome to the Real Estates Home Page!

Hello, agent testf.

Our business hours are 9am-7pm for 7 days a week. Select one of the following options to begin your property search.

[Click to Search for Properties](#)

[Click to View Your Meetings](#)

[Click to Mark Properties As Sold](#)

[Click to Upload Property Image](#)

Upon successful login, the user will be directed to a home page where they can select from several options. These options will vary depending on the type of user account.

- A buyer account can search for properties, book a meeting with an agent, view their bookmarked properties, or view their meetings by clicking on the corresponding links listed on the home page.
- A seller account can either search for properties or add a new property to sell and view their uploaded listings.
- An agent account can search for properties, view meetings, view bookmarked properties, mark properties as sold, or upload a property image.

The Logout button on the top right will sign the user out of their account and redirect them back to the login page.

### **Search Properties Page:**

**Search many listings from trusted real estate agents.**

Select a property type: Residential ▾

City:

Min Price:  ▾

Max Price:  ▾

Bedrooms:  ▾


Bathrooms:  ▾

Users will be required to fill in several parameters to search for a property. From a dropdown menu, they will need to select a property type, a minimum price, maximum price, as well as the number of bedrooms and bathrooms. A city name will also need to be entered as input in the appropriate field. However, if users have no preference in a certain parameter, they may leave that field blank and the system will filter all houses we have that match your other specified requirements. After the user finishes selecting their desired criteria, they can press the “Search” button and all properties satisfying those criteria will be displayed on the screen. From there, the user can choose to view more details about each of the properties that appear in the search results.

## Book a Meeting Page (for buyers only):

### Contact Us

Leave us a message to book an appointment for an online meeting.  
Please note the address of the property that you wish to book a meeting for inquiries on.



Agent (ID):  
1

Date:

Time:  
9:00am

Property ID:  
1

Would you like the meeting to include a tour? ☐ Yes ☐ No

Subject:

Buyers can schedule an online ZOOM meeting with a real estate agent for additional inquiries about their selected property. Upon clicking the link from the home page, they will be navigated to a form in which they can fill out details to book a meeting. All fields must be completed but the “Subject” field is optional. The user must select an agent ID, a date, time, property ID and select the option to include a tour or not. They can then click the “Submit” button to book this meeting, which will be displayed in the View Meetings page.

## View Meetings Page (for buyers and agents only):

### My Meetings:

Meeting ID: 1

Agent ID: 2

Buyer Email: testb

Date: 2021-04-17

Time: 16:00:00

Tour ID: 1

Property ID: 1

Message: i would like to book a meeting!

Buyers can be directed to this page to view their meetings. If they have booked a meeting, the meeting details will be displayed. If they wish to cancel the meeting, they




can do so by clicking on the “Cancel Meeting” button located below the meeting information. This will remove the meeting from the screen and database to indicate the cancellation. Real estate agents will also have access to this view meetings page where they can cancel the meeting on their end too.

### View Bookmarked Properties Page (for buyers only):

My Bookmarks	
Property ID: 1	
<a href="#">View Listing 1</a>	<a href="#">Remove Listing 1</a>
Property ID: 4	
<a href="#">View Listing 4</a>	<a href="#">Remove Listing 4</a>

Users can bookmark a property during their search and it will be displayed on this page. They can click on the “View Listing” button to return back to the page that displays all the property details of that bookmarked property. If they wish to remove the bookmarked property, the user can click on the “Remove Listing” button.

### Sell Property Page (for sellers only):

Are you looking to sell a property?	
Please fill out the form below so we can upload your property to the system.	
	Property Type: <input type="radio"/> Residential <input type="radio"/> Commercial
	Address: <input data-bbox="802 1402 1367 1438" type="text" value="The address of the property..."/>
	Neighbourhood: <input data-bbox="802 1486 1367 1522" type="text" value="The neighbourhood of the property..."/>
	City: <input data-bbox="802 1570 1367 1606" type="text" value="The city of the property..."/>
	Zip Code: <input data-bbox="802 1654 1367 1690" type="text" value="The ZIP code of the property..."/>
Estimated Value: <input data-bbox="802 1738 1367 1774" type="text" value="The estimated value of the property..."/>	

Sellers can upload a new property to the database by filling out the form. The seller must complete all fields to successfully upload the new property. The fields to be

completed include the property type, address, neighbourhood, city, ZIP code, estimated value, square footage, number of bedrooms, and number of bathrooms. After clicking the “Submit” button, the property will be stored in the database.

### **View Listings Page:**

My Listings	
Property ID: 1	
Address: 220 Hawkwood Boulevard NW	
<a href="#">View Listing 1</a>	<a href="#">Delete Listing 1</a>
Property ID: 2	
Address: 23 Applecrest Court SE	
<a href="#">View Listing 2</a>	<a href="#">Delete Listing 2</a>
Property ID: 3	
Address: 5, 1922 9 Avenue SE	
<a href="#">View Listing 3</a>	<a href="#">Delete Listing 3</a>

Sellers can also view the property listings they have uploaded to the system through the Sell Property page. If they have not posted any listings, nothing will be displayed. The “View Listing” button will direct the seller to the page that presents all details on the selected property to the buyer. The “Delete Listing” button will remove the listing from the database and it will no longer appear on this screen or show up when you try to search for that property.

### **Mark Properties as Sold Page (for agents only):**

Mark Property As Sold
Property ID: 1
Address: 220 Hawkwood Boulevard NW
<a href="#">Mark Property 1 As Sold</a>

A real estate agent can mark their properties that have been bought by a buyer. If a buyer wishes to buy a property listed by that agent, the agent can use this page to mark it as sold and it will no longer be visible if other buyers try to search for it because it has been removed from the database.

### **Upload Property Image Page (for agents only):**

**Upload Property Image**

Property ID:   No file chosen

When a property listing is uploaded to the system, there will be a default photo used. This default photo will be displayed for the property in the property search results, as well as in the property details pages.

A real estate agent can upload a new property image by selecting the property ID to indicate which listing they want to change the photo on and then clicking the upload button and selecting a JFIF image file from their device.