

COS30043 – Interface Design and Web Development

Task 6.2D-HD CUSTOM WEB APPLICATION



Unit: COS30043

Name: Le Gia Hoang An

Student ID: 104789808

Introduction

This custom project is about a Real Estate Listing Website, where a user can view all the properties that are currently being listed. The details about the real estate agent team are also available for guest users to view. When user is logged into the website as administrator, they get an additional tab for authorised users only, where they can see all data about the properties and make any changes to that list. This project will be show-casing the implementation of high-quality features come from effectively integrating and utilising frameworks.

Requirements

Technical Requirements

This web application has a responsive design that is supported by Bootstrap's framework to organise and structure the layout of the web page's content. This also enable the web page to follow mobile-first approach that makes it responsive to various device sizes. The project strictly follows camel case and kebab case coding conventions, to make the code more readable. By utilising VueJS frameworks, features such as vue-routers and vue-paginate also contribute to help user navigate the web page and keeping the data table more compact.

Default directives such as v-model, v-if, v-bind, and v-on are also implemented to make the web application more reactive and interactive. These features are useful for dynamically render data based on preset conditions. Furthermore, custom directives can also be created in the main JavaScript file that connects to the vue application. For example, I was able to create custom text colouring based on dynamic data fetched and rendered by backend system.

Dynamic data rendering is powered by custom Fetch APIs that connects to a MySQL database, which uses PHP backend system to post and retrieve data. The fetch APIs are called when the web page first open, using methods and computed properties from VueJS framework.

For the user with admin access, they can insert, update, and delete data using forms. Each form is incorporated with VueJS v-model directives and Bootstrap layout. The form also comes with its own data validation process that checks for errors such as empty input fields or incorrect data format.

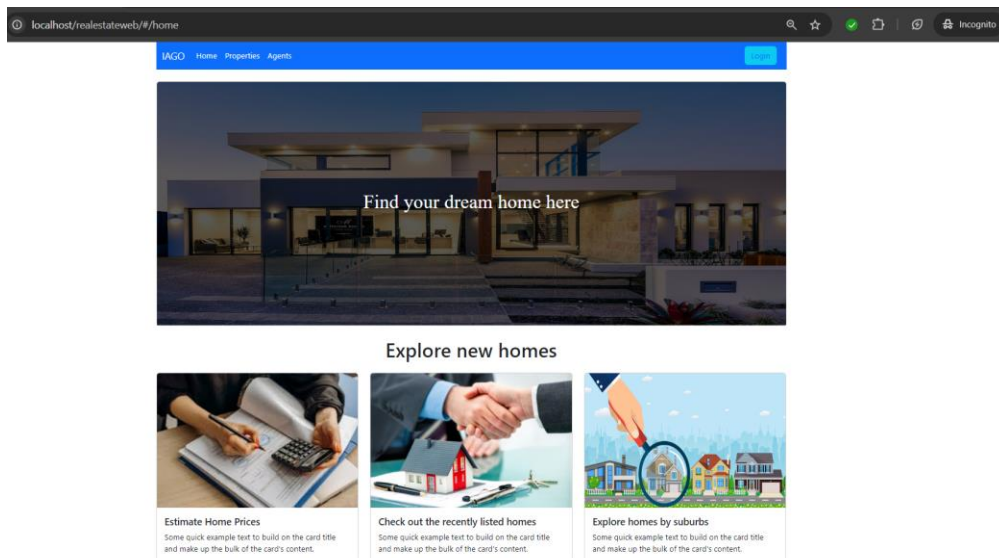


Figure 1: Implementation of Bootstrap Grid System for responsive UI

```

1  const Properties = {
38  template: `
54      <div class="form-check">
55          <input v-model="typeRadio" class="form-check-input" type="radio" name="type" id="status" value="Available" />
56          <label for="status">Available</label>
57      </div>
58
59      <div class="form-check">
60          <input v-model="typeRadio" class="form-check-input" type="radio" name="type" id="status"
61              value="Under Contract">
62          <label for="status">Under Contract</label>
63      </div>
64  </div>
65  </div>
66
67  <br><br>
68
69  <div v-for="house in getFilteredAndPaginatedProperties" :key="house.PropertyID">
70      <div class="card mb-3 rounded-5">
71          <div class="row g-0">
72              <div class="col-md-4">
73                  
74              </div>
75              <div class="col-md-8">
76                  <div class="card-body">
77                      <h5 class="card-title">ListingPrice: &#36;{{numberWithCommas(house.ListingPrice)}}</h5>
78
79                      <p><small>Status: <strong v-status="house.Status">{{house.Status}}</strong></small></p>

```

Figure 2: Implementation of directives, camel case and kebab case in HTML5

Functional Requirements

The web application implements features account registration and login function. The user and registered users can both search for property they are looking for, with additional support from filter search to refine results. Registered users will have the ability to insert, update and delete persistent data that is stored in MySQL database through PHP Fetch API. They can also give a like to a real estate agent that they like.

Innovative Features

Custom directives are implemented for colouring text based on the dynamic data that are generated from v-for directive. Some contents on this web application are hidden to guest users. For example, the Dashboard menu item on the navigation bar will only show up when the user is logged in where they can perform data insert, update, and delete. The property listing in View tab automatically refreshes every three seconds to keep the data updated in real time. Liking button for social feature is only available for registered users.

As previously mentioned, registered users can give a like to a real estate agent. However, they can only like an agent once, otherwise they would just unlike that agent. This is due to each real estate agent has a column that stores an array of registered username that liked their profile. Clicking on the like button again will remove their name from that column array.

When new data on a new property gets inserted or updated, the image of the property will get stored into a local image folder within the project and only the image link path is saved into the database. The name of the image was given a unique ID to prevent duplicate names, in case two or more users are using the same image.




IAGO Home Properties Agents Dashboard Logout							
Dashboard (Authorised user only)							
View Insert Update Delete							
PropertyID	Location	Listing Price	Listing Date	Attributes	Status	Image	Agent
1	<ul style="list-style-type: none">Address: 31 Ludington CenterCity: San DiegoState: CaliforniaPost Code:92170	\$202,016.58	2022-01-05	<ul style="list-style-type: none">Square Footage: 3707 m²Bedrooms: 6Bathrooms: 4Garages: 3	Under Contract		Danni Korda
2	<ul style="list-style-type: none">Address: 25504 Carey ParkCity: ChicagoState: IllinoisPost Code:60641	\$746,717.32	2021-01-13	<ul style="list-style-type: none">Square Footage: 4393 m²Bedrooms: 5Bathrooms: 1Garages: 3	Available		Cindee Raubenheim

Figure 3: Dashboard is only available when user is logged in

IAGO
Home
Properties
Agents
Dashboard
Logout

IAGO Agent Team





Danni Korda

Phone: 106-899-6468

Email: dkorda0@paginegialle.it

License Number: JTHHP5BC4F5423595

 1




Cindee Raubenheim

Phone: 357-890-8700

Email: craubenheim1@businessinsider.com

License Number: WDCYC3HF5AX942987

 0

Previous
1
2
3
Next

Figure 4: Liking feature is only available when user is logged in

```

39      ///Image Data Processing
40      $fileDestination = '';
41      if (isset($_FILES['Image'])) {
42          $file = $_FILES['Image'];
43          $fileName = $file['name'];
44          $fileTmpName = $file['tmp_name'];
45          $fileSize = $file['size'];
46          $fileError = $file['error'];
47          $fileType = $file['type'];
48          $fileExt = explode('.', $fileName);
49          $fileActualExt = strtolower(end($fileExt));
50          $allowed = array('jpg', 'jpeg', 'png');
51
52          //This part will save the uploaded file onto the ./css/img folder
53          //within this project source code folder.
54          if (in_array($fileActualExt, $allowed)) {
55              if ($fileError === 0) {
56                  if ($fileSize < 10000000) { // Size should be in bytes, 10MB = 10000000 bytes, images are
57                      $fileNameNew = uniqid('', true) . "." . $fileActualExt;
58                      //XAMPP folder destination
59                      $uploadDir = __DIR__ . '/../css/img/';
60                      // Make sure the directory exists
61                      if (!is_dir($uploadDir)) {
62                          mkdir($uploadDir, 0777, true); // Create directory if it doesn't exist
63                      }
64                      $fileDestination = $uploadDir . $fileNameNew;
65                      move_uploaded_file($fileTmpName, $fileDestination);
66                  } else {
67                      echo "File too big!";

```

Figure 5: PHP code for storing uploaded image into a local folder

```

const ViewProperties = {
  components: {
    paginate: VuejsPaginateNext,
  },
  data() {
    return {
      message: "Loading...",
      isLoading: false,
      perPage: 6,
      currentPage: 1,
      properties: [],
    };
  },
  created() {
    this.loadDataView();
    // Auto Refresh View Data Table every 3 seconds
    setInterval(this.loadDataView, 3000);
  },
  template: `
    <p v-if="isLoading">{{message}}</p>
    <table class="table table-responsive">
      <thead>
        <tr>
          <th scope="col">PropertyID</th>
          <th scope="col">Location</th>
          <th scope="col">Listing Price</th>
          <th scope="col">Listing Date</th>
          <th scope="col">Attributes</th>
          <th scope="col">Status</th>
          <th scope="col">Image</th>
        </tr>
      </thead>
    </table>
  `

```

Figure 6: Property Listing gets automatically refresh every 3 seconds with setInterval function