**IMAGE SEARCH**

**Requirements:**

Provide the capability to search HCL Commerce Catalog using image search on React Stores.

**User Flow:**

There are two ways image search can be done.

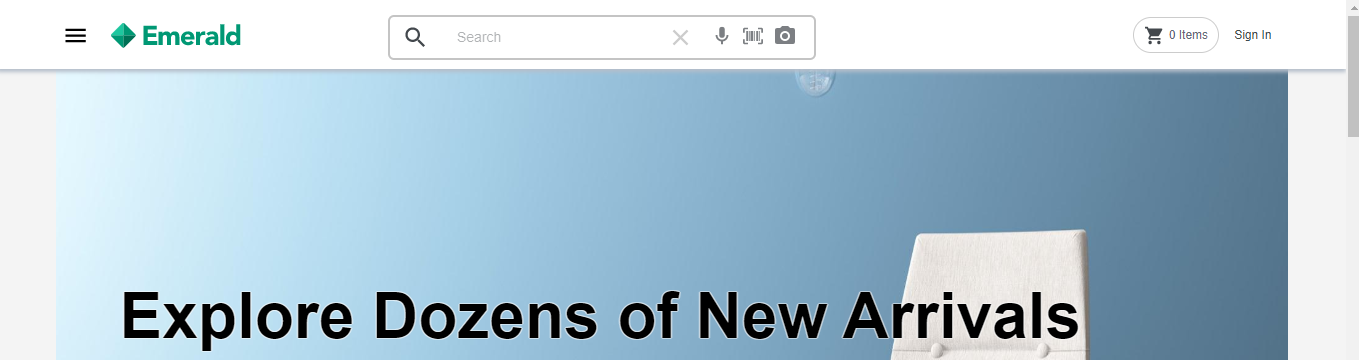
1. By uploading an image.
2. By opening the web cam and capturing the image.

For uploading we have used javascript input tag to accept only image file file.

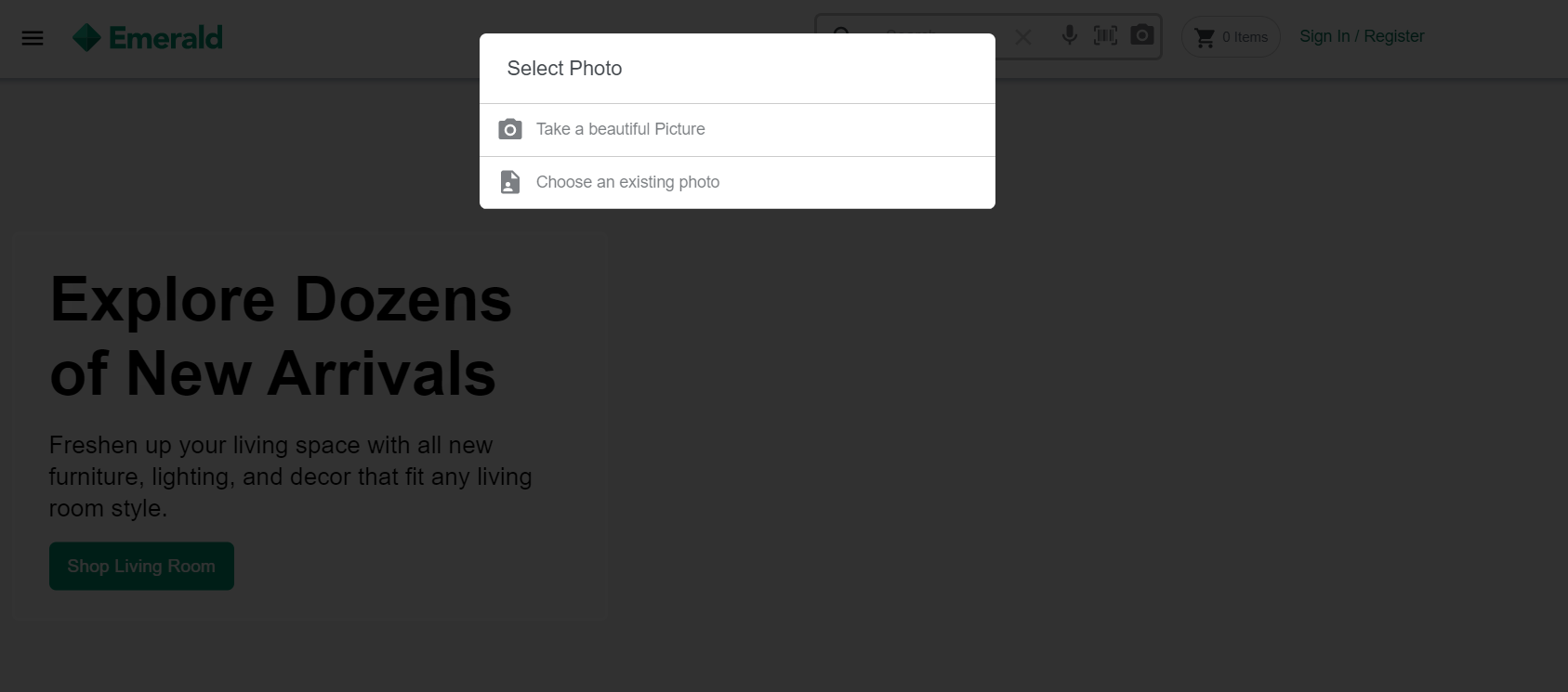
For capturing the image we have used the react-web cam feature as below.

import Webcam from "react-webcam";

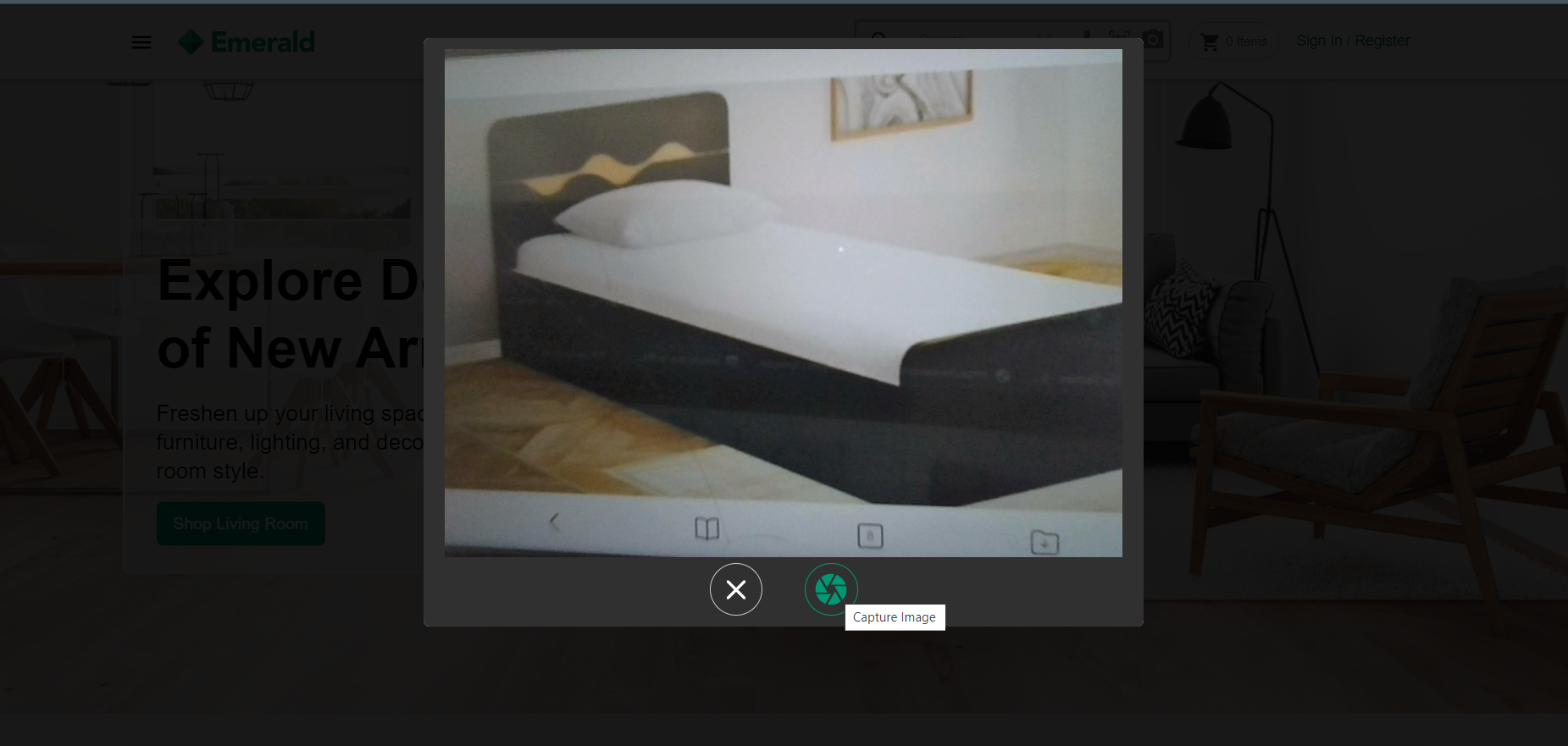
The search bar appears as a below on load.



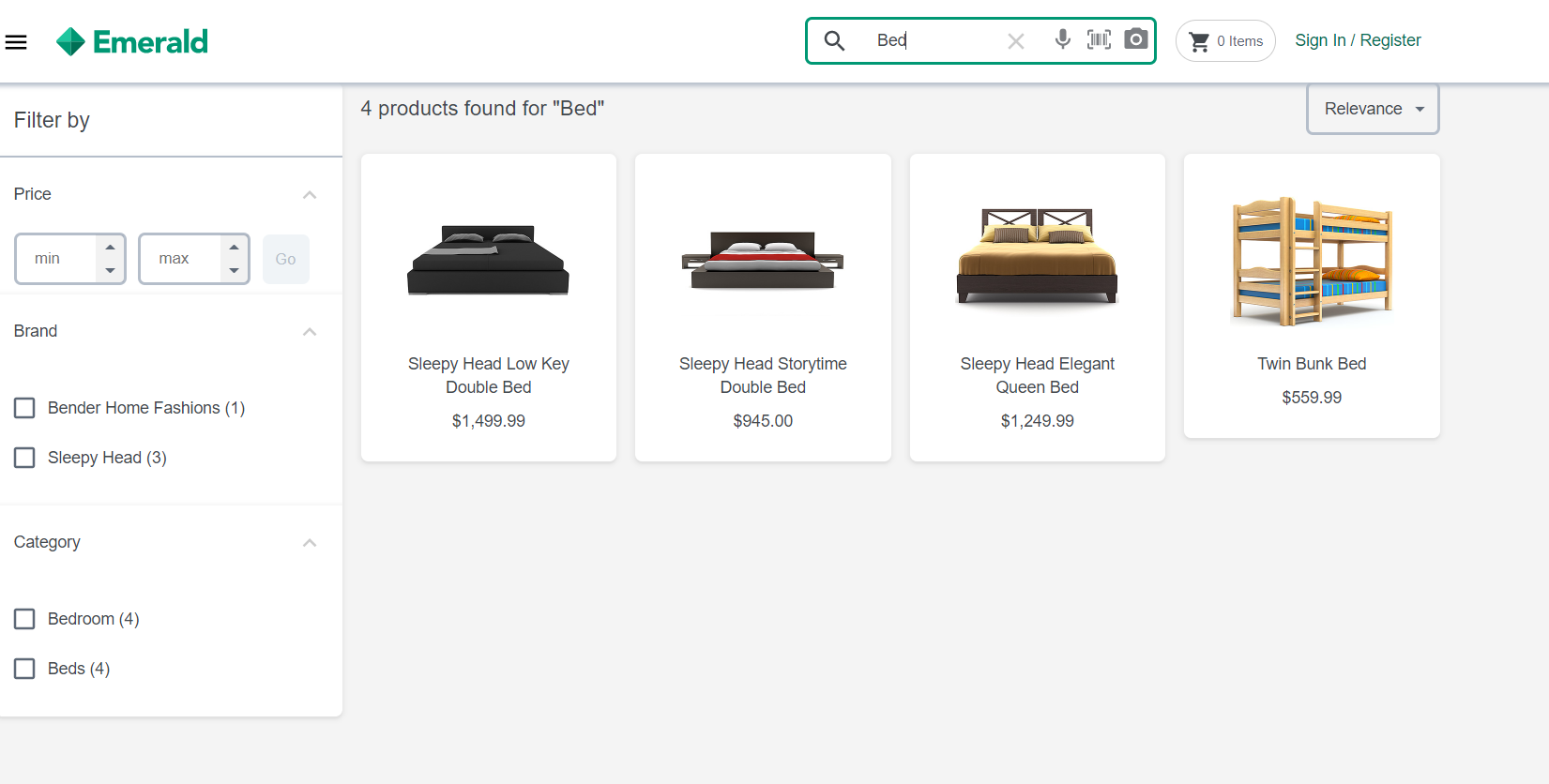
On click of camera icon the window is opened as below where we see 2 options.

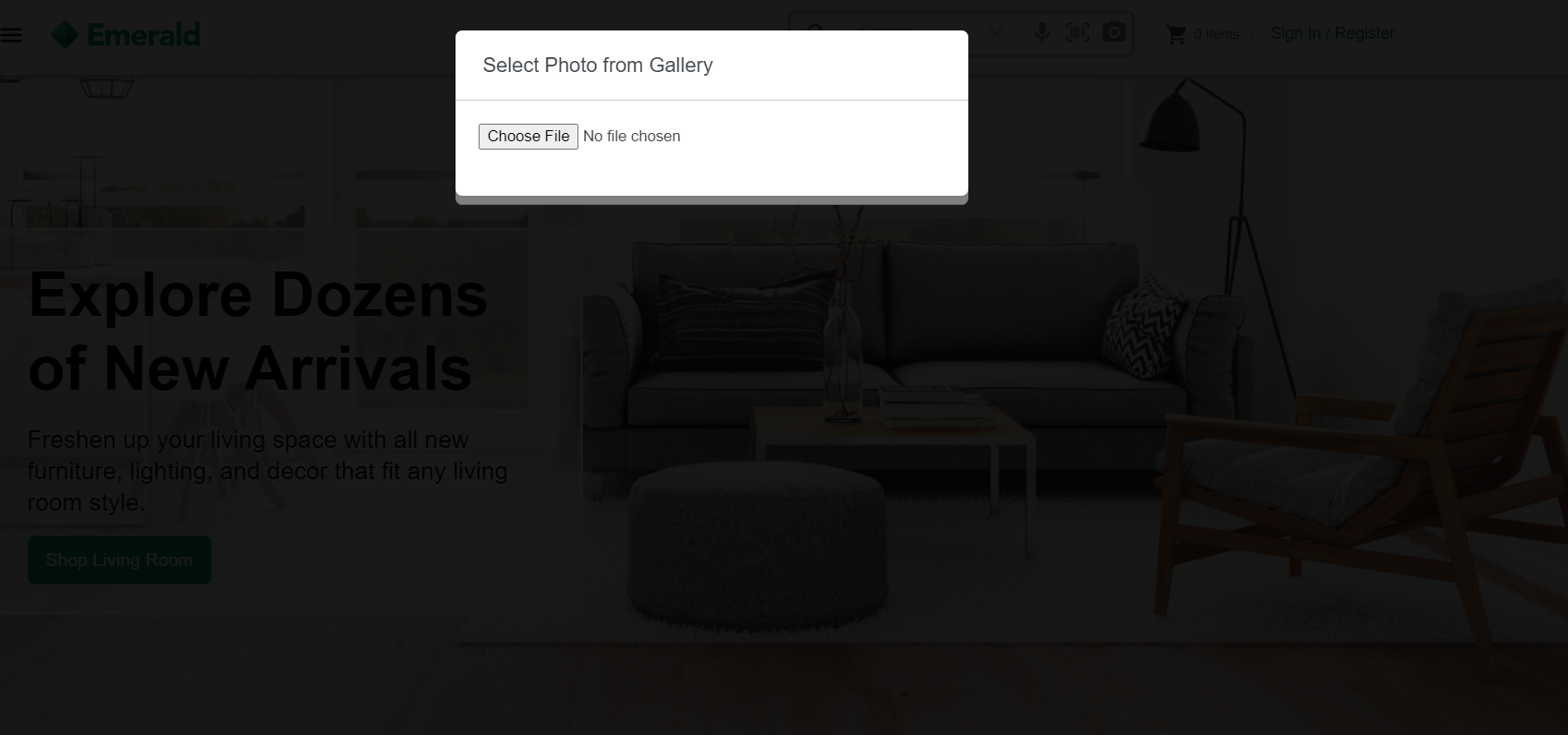


Select “Take a Beautiful Picture” .Below Popup window will open with camera and user can click an image.

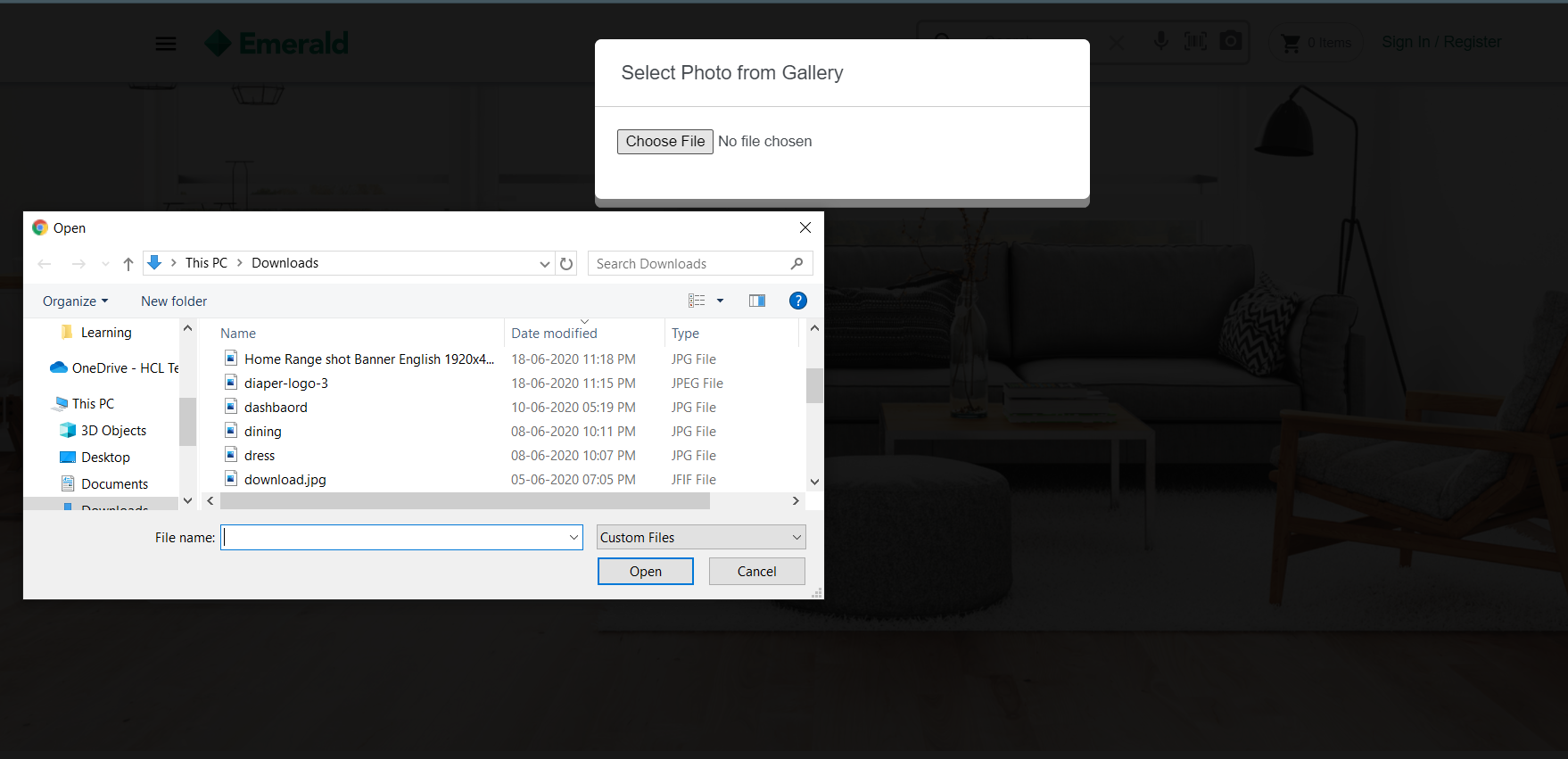


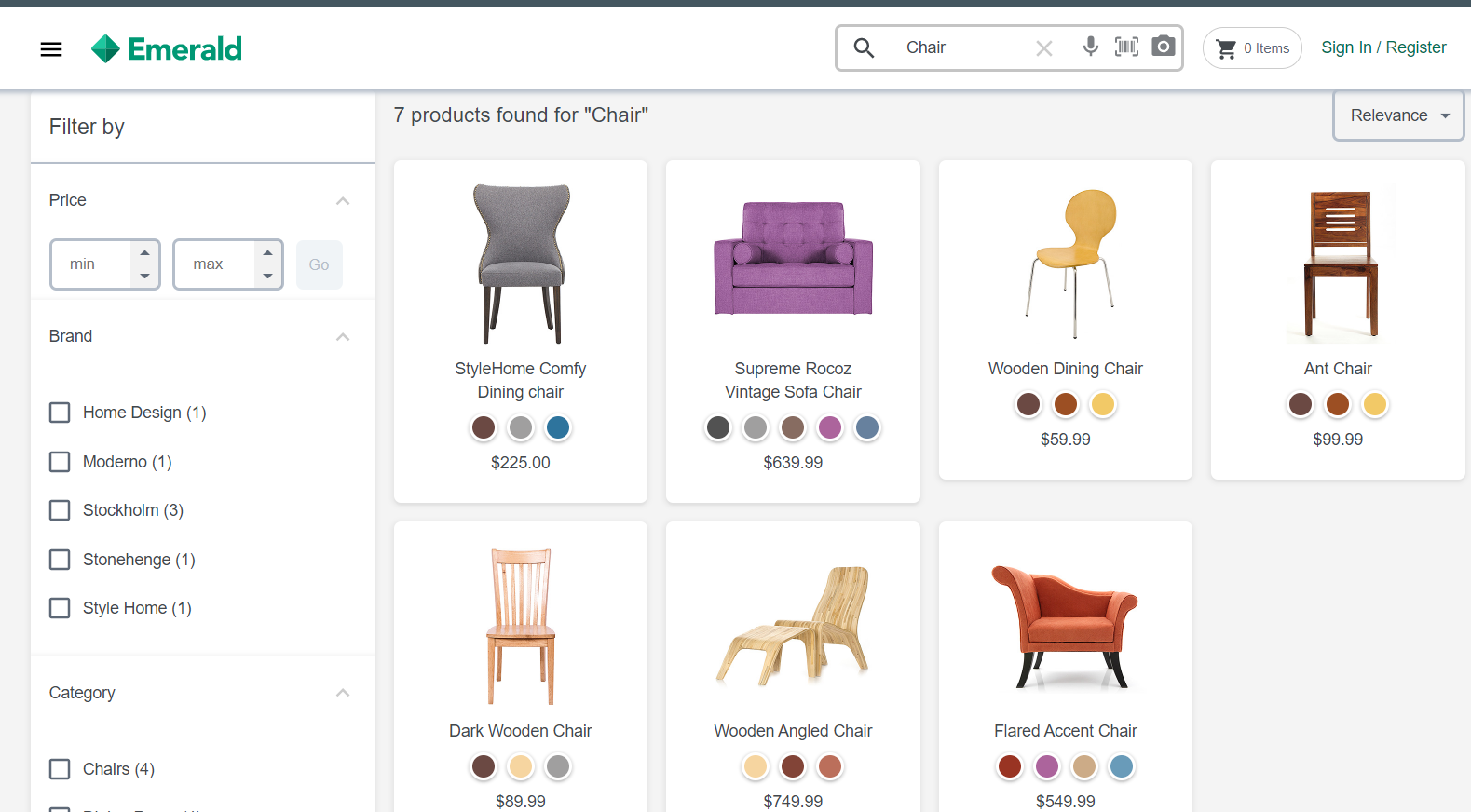
We have captured the image of a bed which is returned by the Google Image Search Service.



There is another option of uploading an image where user can upload an image.

It returns me the exact match of the image.





Note :

The library “react-webcam “used for taking pictures doesn't support in IOS chrome as this library uses HTML5 Media API “getUserMedia” which is not supported by chrome in IOS.

**Steps to include the Image search scanner in your project:**

1. You need to install the react web cam in your project as a dependency.

Complete usage is given at <https://www.npmjs.com/package/react-webcam>

It can be installed using npm.

> npm install react-webcam –save

Once installation is done. Verify the entry in your package.json file.

1. Include the search-types folder in src/components/widgets folder. The below two files are required for image search from this searc-type folder.

- search-types.tsx

- imagetotext.tsx

1. Below changes need to add in the SearchBar.tsx file present in src/components/widgets/search-bar folder

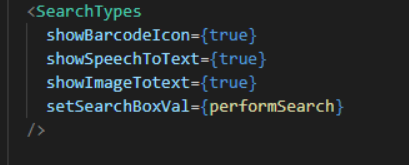
We have created a Search Type component to specify the search type for the user.

User can configure this according to his choice

Import this file in the SearchBar.tsx file.

import { SearchTypes } from "../Search-types/search-types";

This component can be used as below.





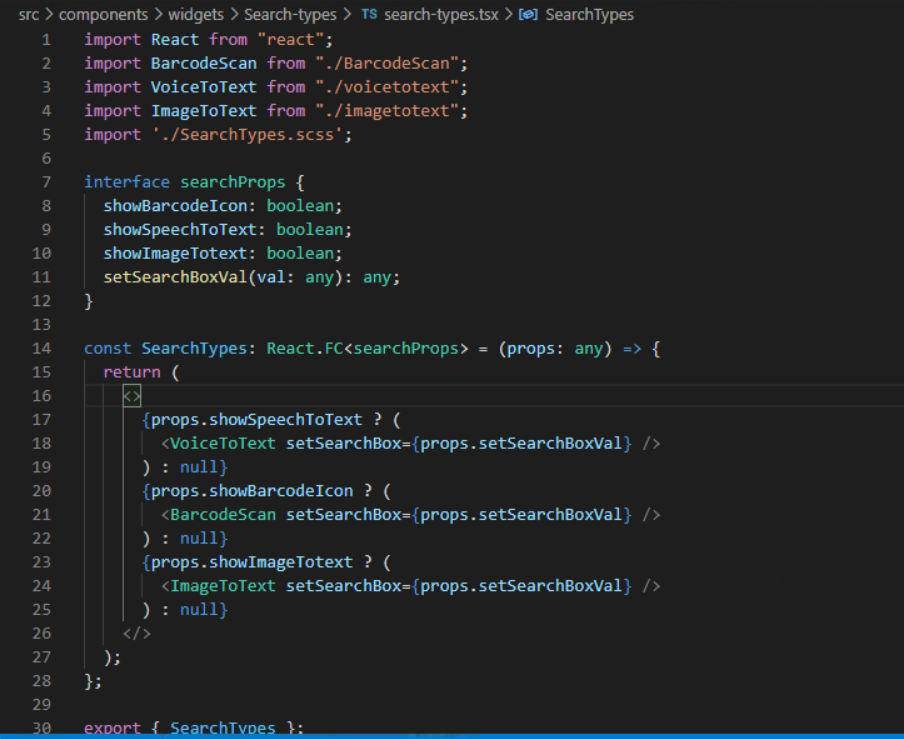
showBarcode Iconà used for barcode scanning.

showSpeechToText à used for Voice transcribe.

showImageToText à used for Image search.

setSearchBoxValà used to set the value of the search result.

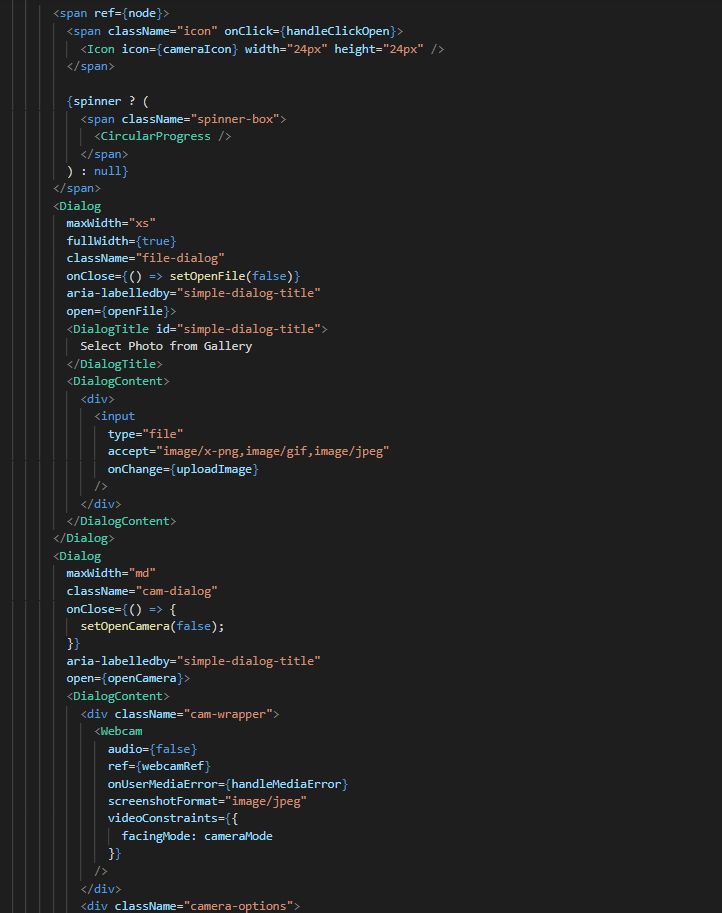
1. The Search Types component looks like below.



In this document we will see only the ImageToText Component.

1. ImagetoText.tsx file looks like below:

Partial snippet (whole source code is shared in codebase folder)



The clickon the capture button captures the image and calls the Google Image Service.

**Google Image API is used to translate the audio to text.**

It takes the base 64 encoded image as an input and gives the translated text as a output.

We have created the firebase api to call the google speech API.

The call to the firebase API is placed in the voiceImageTranscribeService.

const imageToText = async base64bytes => {

    try {

      const res = await voiceImageTranscibeService.getImageTranscibetext(

        base64bytes

      );

      console.log("RESPONSE RECEIVED: ", res);

      res.data.responses

        ? props.setSearchBox(

            res.data.responses[0].localizedObjectAnnotations[0].name

          )

        : props.setSearchBox("No Data Found...");

      setSpinner(false);

    } catch (err) {

      console.log("ERROR: ", err);

      setSpinner(false);

    }

  };

      console.log("RESPONSE RECEIVED: ", res);

      res.data.results

        ? props.setSearchBox(res.data.results[0].alternatives[0].transcript)

        : props.setSearchBox("No Data Found...");

      setSpinner(false);

    } catch (err) {

      console.log("ERROR: ", err);

      setSpinner(false);

    }

  };

1. Include the voiceImageTranscribeService.js file in src/\_foundation/apis/search folder

The code in the service looks like below.

getImageTranscibetext(imageBytes): AxiosPromise<any> {

    let requestOptions: AxiosRequestConfig = Object.assign({

      data: { imageBytes },

      url: IMAGE\_URL,

      method: "post"

    });

    return Axios(requestOptions);

  }

};

const IMAGE\_URL =

  "https://us-central1-emerald-a9fa8.cloudfunctions.net/app/image-recognition";

The above service returns the transcribed word of the user’s image uploaded.