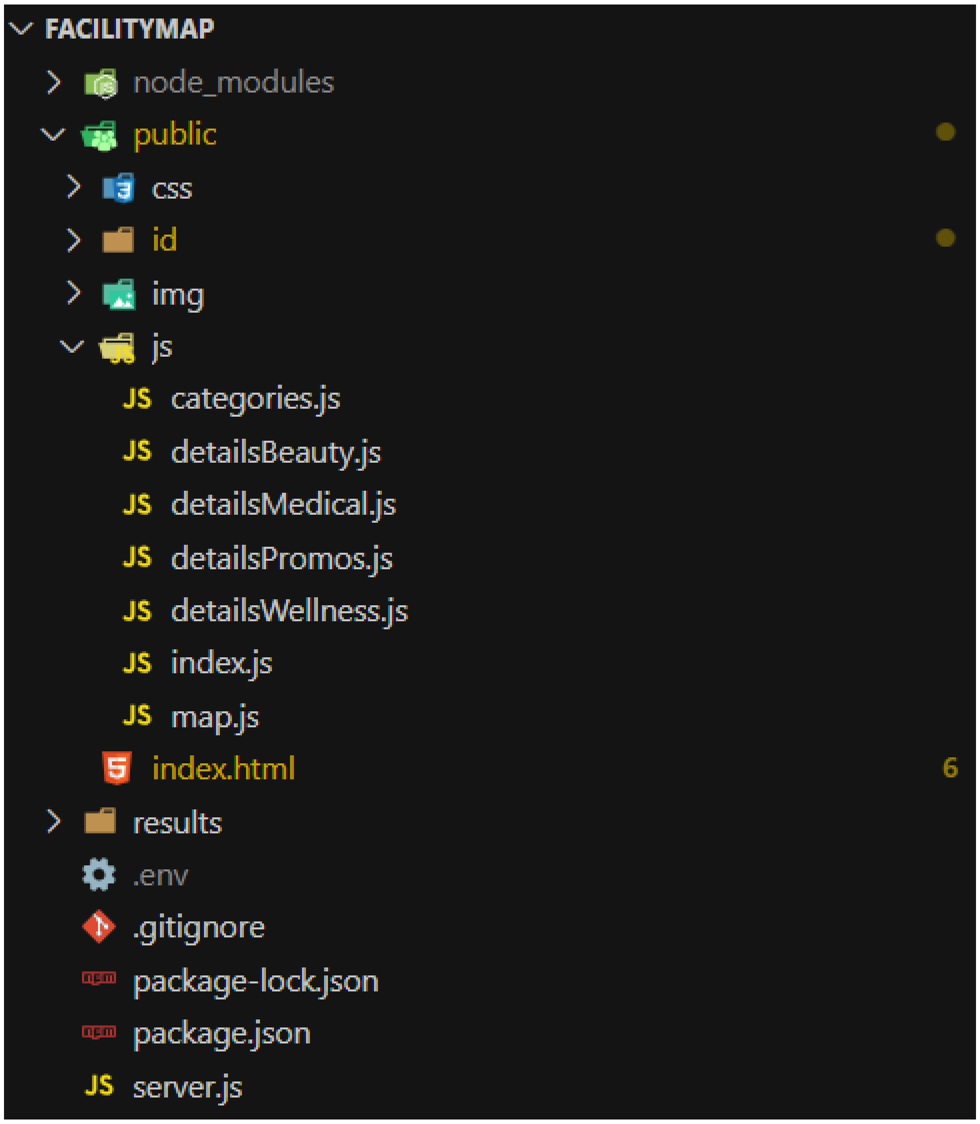
Medical Hub Map Maintenance Guide

This guide concerns the addition of,

1. *places*
2. *categories,*
3. *subcategories,*
4. *Promos, &*
5. *reviews*



```
FACILITYMAP
  node_modules
  public                      ●
    css
    id                        ●
    img
    js
      categories.js
      detailsBeauty.js
      detailsMedical.js
      detailsPromos.js
      detailsWellness.js
      index.js
      map.js
    index.html                6
  results
  .env
  .gitignore
  package-lock.json
  package.json
  server.js
```

Image I. *File Structure*

I. Basics

Purpose of This Guide:

This guide is designed to help those who are not experienced with JavaScript update/maintain the Medical Hub Map easily. However, it’s still a good idea for someone who knows JavaScript to review the changes before they are put into use to make sure everything works correctly.

About the Map:

The Medical Hub Map uses the Google Maps API to show a basic map and provide directions from a user's location. Everything else on the map, like markers, categories, and promotions, comes from files within the project. To keep the project safe and working properly, you should not change most of the files.

A caution of warning as due to the nature of Google Maps API, calling the API will incur costs. So for any maintenance done, please be aware of such a fact and consult with those incharge of the program before doing anything.

What You Can Edit:

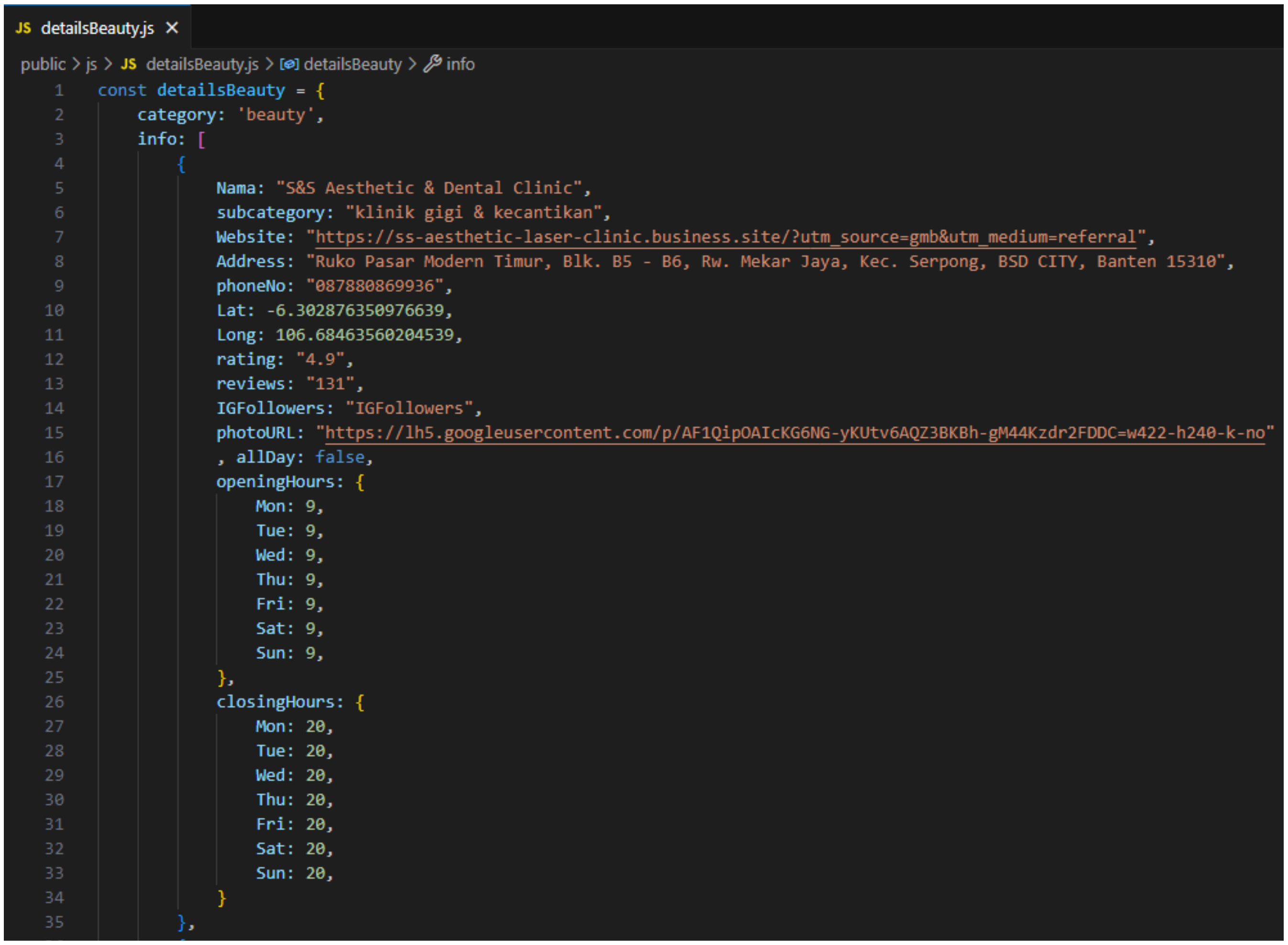
If you need to add categories, subcategories, locations, promotions, or reviews, you can edit certain files. These files are in the ‘js’ folder, which is inside the ‘public’ folder of the project. Make sure to be careful when editing these files. Always create a backup before making any changes.

Important Note:

If any issues come up while you're making changes, you must inform the IT team right away.

II. Adding Places

The simplest action that can be done is the addition of places to a predetermined category. Other than determining what category a location should be counted in, the steps to add a place involves adding it to the file of said category.



JS detailsBeauty.js ×

public > js > JS detailsBeauty.js > detailsBeauty > info

```
const detailsBeauty = {
    category: 'beauty',
    info: [
        {
            Nama: "S&S Aesthetic & Dental Clinic",
            subcategory: "klinik gigi & kecantikan",
            Website: "https://ss-aesthetic-laser-clinic.business.site/?utm_source=gmb&utm_medium=referral",
            Address: "Ruko Pasar Modern Timur, Blk. B5 - B6, Rw. Mekar Jaya, Kec. Serpong, BSD CITY, Banten 15310",
            phoneNo: "087880869936",
            Lat: -6.302876350976639,
            Long: 106.68463560204539,
            rating: "4.9",
            reviews: "131",
            IGFollowers: "IGFollowers",
            photoURL: "https://lh5.googleusercontent.com/p/AF1QipOAIcKG6NG-yKUtv6AQZ3BKBh-gM44Kzdr2FDDC=w422-h240-k-no"
            , allDay: false,
            openingHours: {
                Mon: 9,
                Tue: 9,
                Wed: 9,
                Thu: 9,
                Fri: 9,
                Sat: 9,
                Sun: 9,
            },
            closingHours: {
                Mon: 20,
                Tue: 20,
                Wed: 20,
                Thu: 20,
                Fri: 20,
                Sat: 20,
                Sun: 20,
            }
        },
```

Image II. Example of a File of the ‘Beauty’ Category. Filename: detailsBeauty.js

When adding new places, you must format them within brackets, as shown in Image II. Each place has several attributes:

- Nama
- Subcategory
- Website

- Address
- phoneNo
- Lat (Latitude)
- Long (Longitude)
- rating
- reviews
- IGFollowers
- photoURL
- allDay
- openingHours
- closingHours
- reviewsText

Required Attributes:

To add a place, you must fill in the following attributes:

1. Nama: The name of the place. Make sure it's accurate for easy reference.

2. Subcategory: The subcategory of the place. If it already exists, copy it exactly from the categories.js file to ensure consistency.

3. Address: The location's address. It doesn't need to be very detailed since the exact location is set by latitude and longitude.

4. Lat (Latitude): The latitude coordinate. To find it, right-click on the location in Google Maps and use the first number. (See Image III)

5. Long (Longitude): The longitude coordinate. Right-click on the location in Google Maps and use the second number. (See Image III)

6. rating: The place's rating on Google Maps (out of five stars). If unavailable, remove this attribute.

7. openingHours: The opening time in 24-hour format. For example, 9 for 9 am. (See Image IV for more details).

8. closingHours: The closing time in 24-hour format. For example, 18, for 6 pm. If the place is closed on a certain day, set the same time for both opening and closing. (See Image IV for more details).

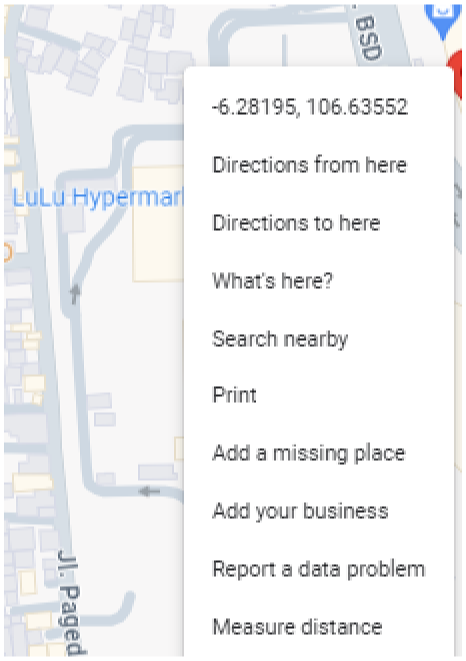
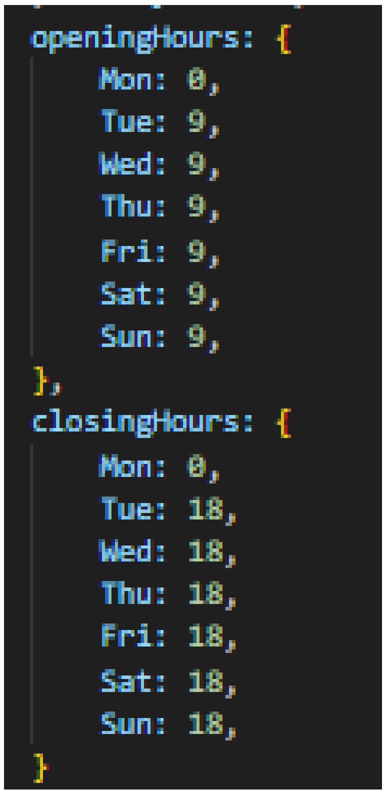


Image III. Latitude and Longitude. In this example, -6.28195 is longitude, and 106.63552 is longitude.



```
openingHours: {
    Mon: 0,
    Tue: 9,
    Wed: 9,
    Thu: 9,
    Fri: 9,
    Sat: 9,
    Sun: 9,
},
closingHours: {
    Mon: 0,
    Tue: 18,
    Wed: 18,
    Thu: 18,
    Fri: 18,
    Sat: 18,
    Sun: 18,
}
```

Image IV. An example of the usage of openingHours and closingHours. Here, it says the location is open Tuesday to Sunday from 9 to 18 o'clock. But Closed on Mondays as indicated by Mon in openingHours and closingHours using the same number of '0'.

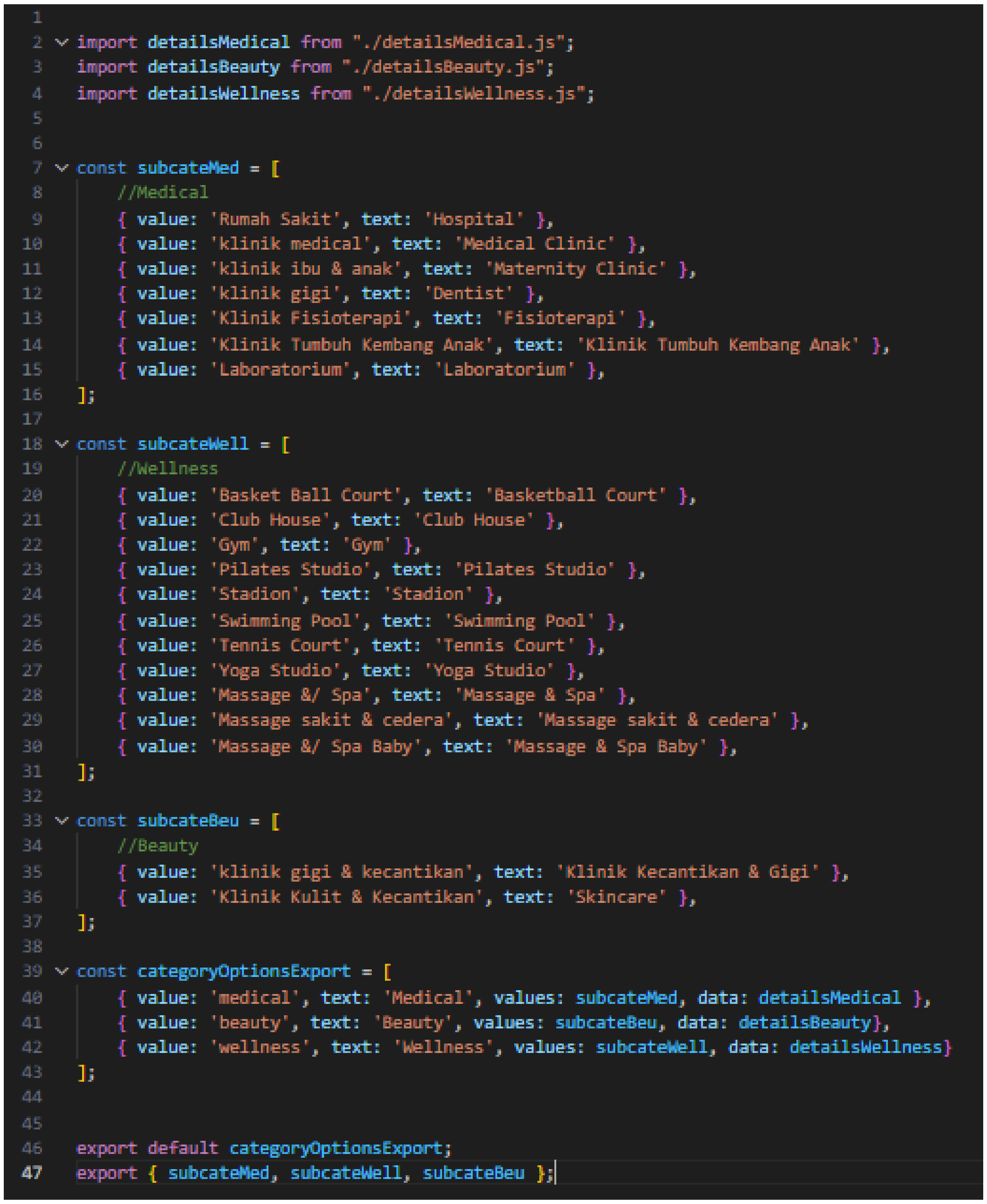
Optional Attributes:

These attributes are not required but are recommended if the information is available:

- Website: The place’s website link. (Instragram page is also allowed, as long it is a link)
- phoneNo: The contact phone number.
- reviews: The number of reviews on Google Maps. Remove if unavailable.
- IGFollowers: The number of Instagram followers for the place’s account.
- photoURL: A link to a photo of the location (must end in .png, .jpg, or .jpeg).
- allDay: Use this to indicate if the place is open 24 hours a day.
- Reviews Text: For instructions on adding reviews, see Section VI.

III. Adding Categories.

To add categories, please navigate to the ‘categories.js’ file. Assuming there have been no changes to this file, this is how it will look like.



```
import detailsMedical from "./detailsMedical.js";
import detailsBeauty from "./detailsBeauty.js";
import detailsWellness from "./detailsWellness.js";


const subcateMed = [
    //Medical
    { value: 'Rumah Sakit', text: 'Hospital' },
    { value: 'klinik medical', text: 'Medical Clinic' },
    { value: 'klinik ibu & anak', text: 'Maternity Clinic' },
    { value: 'klinik gigi', text: 'Dentist' },
    { value: 'Klinik Fisioterapi', text: 'Fisioterapi' },
    { value: 'Klinik Tumbuh Kembang Anak', text: 'Klinik Tumbuh Kembang Anak' },
    { value: 'Laboratorium', text: 'Laboratorium' },
];

const subcateWell = [
    //Wellness
    { value: 'Basket Ball Court', text: 'Basketball Court' },
    { value: 'Club House', text: 'Club House' },
    { value: 'Gym', text: 'Gym' },
    { value: 'Pilates Studio', text: 'Pilates Studio' },
    { value: 'Stadion', text: 'Stadion' },
    { value: 'Swimming Pool', text: 'Swimming Pool' },
    { value: 'Tennis Court', text: 'Tennis Court' },
    { value: 'Yoga Studio', text: 'Yoga Studio' },
    { value: 'Massage &/ Spa', text: 'Massage & Spa' },
    { value: 'Massage sakit & cedera', text: 'Massage sakit & cedera' },
    { value: 'Massage &/ Spa Baby', text: 'Massage & Spa Baby' },
];

const subcateBeu = [
    //Beauty
    { value: 'klinik gigi & kecantikan', text: 'Klinik Kecantikan & Gigi' },
    { value: 'Klinik Kulit & Kecantikan', text: 'Skincare' },
];

const categoryOptionsExport = [
    { value: 'medical', text: 'Medical', values: subcateMed, data: detailsMedical },
    { value: 'beauty', text: 'Beauty', values: subcateBeu, data: detailsBeauty},
    { value: 'wellness', text: 'Wellness', values: subcateWell, data: detailsWellness}
];


export default categoryOptionsExport;
export { subcateMed, subcateWell, subcateBeu };
```

Image V. Screenshot of Categories.js as of 2024-08-22

When adding a new category, keep in mind that adding categories and subcategories go hand in hand. This section focuses on adding categories; subcategories will be covered in the next section.

Steps to Add a Category:

1. Add a Subcategory Section:
    - Before you can add a new category, you need to set up a subcategory section first. The specifics of this will be explained later.

2. Modify `categoryOptionsExport`:
    - To add a new category, you need to add a line to `categoryOptionsExport` in the format below:

```
{value: "category", text: "Category", values: subcategory, data: detailsCategory},
```

- **`value` and `text` Attributes:**
    - Both `value` and `text` hold the name of the new category you are adding.
    - The `text` attribute is what end-users will see on the website, so make sure it is user-friendly and correctly spelled.
- **`values` Attribute:**
    - This attribute will link to subcategories. Details on how to fill this will be covered in Section IV.
- **`data` Attribute:**
    - This should point to the file containing details of all locations in the category. For easy reference, name this file `detailsCategory`, where "Category" is replaced with the actual category name (e.g., `detailsHotel` for the Hotels category).

3. Import the Details File:
    - Go to the top of the file and add a line to import the details file:

```
import detailsCategory from "./detailsCategory.js";
```

- Replace "Category" with the name of your new category. For example, if adding "Hotels," it would be:

```
import detailsHotel from "./detailsHotel.js";
```

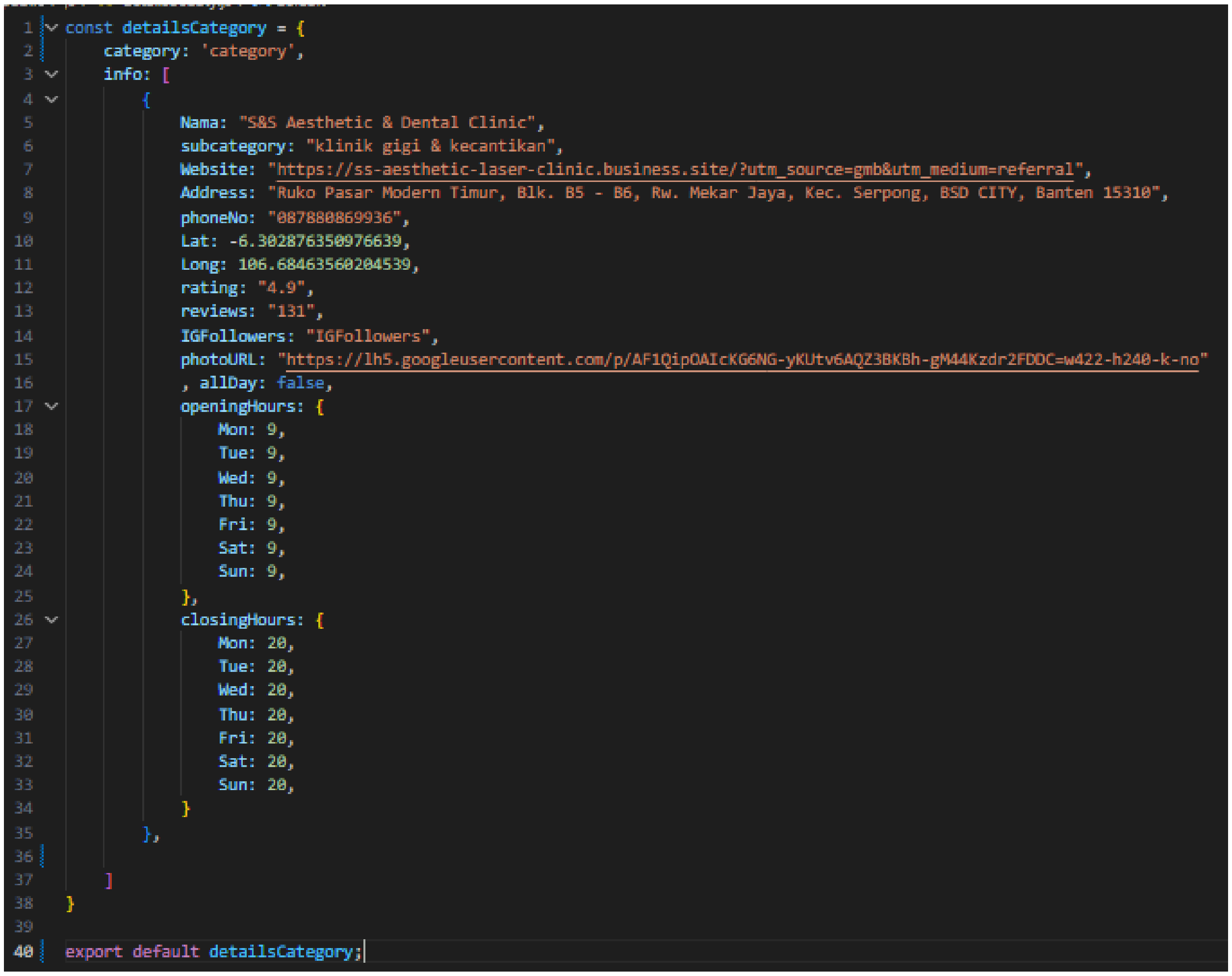
4. Create a New Details File:
   - In the same folder where `categories.js` is located, create a new file named `detailsCategory.js` (replacing "Category" with your category's name, like `detailsHotel.js`).

   - Make sure the file name ends with `.js`.

   - This file will contain all location details for the new category, formatted according to the guidelines in Section II.

How to Create a New File for a Category:

The basic format for these files is straightforward, as shown in Image VI.



```
const detailsCategory = {
    category: 'category',
    info: [
        {
            Nama: "S&S Aesthetic & Dental Clinic",
            subcategory: "klinik gigi & kecantikan",
            Website: "https://ss-aesthetic-laser-clinic.business.site/?utm_source=gmb&utm_medium=referral",
            Address: "Ruko Pasar Modern Timur, Blk. B5 - B6, Rw. Mekar Jaya, Kec. Serpong, BSD CITY, Banten 15310",
            phoneNo: "087880869936",
            Lat: -6.302876350976639,
            Long: 106.68463560204539,
            rating: "4.9",
            reviews: "131",
            IGFollowers: "IGFollowers",
            photoURL: "https://lh5.googleusercontent.com/p/AF1QipOAIcKG6NG-yKUtv6AQZ3BKBh-gM44Kzdr2FDDC=w422-h240-k-no"
            , allDay: false,
            openingHours: {
                Mon: 9,
                Tue: 9,
                Wed: 9,
                Thu: 9,
                Fri: 9,
                Sat: 9,
                Sun: 9,
            },
            closingHours: {
                Mon: 20,
                Tue: 20,
                Wed: 20,
                Thu: 20,
                Fri: 20,
                Sat: 20,
                Sun: 20,
            }
        },

    ]
}

export default detailsCategory;
```

Image VI. The simplest form detailsCategory.js make take

When you create a new file for a category, you need to start with a specific template. Here's what you should include:

```
const detailsCategory = {
  Category: 'category',
  Info: [
    {
      // Place details go here
    },
  ],
}
```

```
export default detailsCategory;
```

Steps to Set Up the File:

1. Modify the Template:
    - Change `detailsCategory` to the name of your category.

    - Change `'category'` to match the category name.

For example, If you are creating a file for hotels, it should look like this:

```
const detailsHotel = {
    Category: 'hotel',
    Info: [
      {
        // Place details go here
      },
    ],
  }

  export default detailsHotel;
```

2. Add Place Details:
    - Once you have set up the file, you can start adding places according to the format provided in Section II of the guide.

After completing these steps, your new category is ready to be used.

IV. Adding Subcategories

To add a subcategory, please navigate to ‘categories.js’.

1. Understand the Format:
    - When adding a new category, the basic format is:

```
{value: "category", text: "Category", values: subcategory, data: detailsCategory},
```

- The `subcategory` in this format refers to an array you will create in the same file.

2. Create a Subcategory Array:
    - You need to create an array for the subcategories. The array name can be anything, but to keep it clear, use the format `subcateCategory`, replacing "Category" with the actual category name.

For Example, If you’re adding subcategories for hospitals, name the array `subcateHospital`.

- Use the following template to create your array:

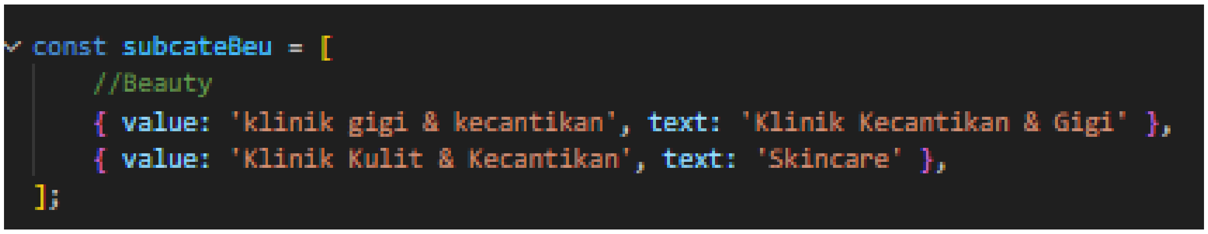
```
const subcateCategory = [
  {value: 'subcategoryid', text: 'subcategoryname'},
];
```

Replace `Category` with the new category name (e.g., `subcateHospital`), `subcategoryid` with a unique identifier for the subcategory, and `subcategoryname` with the name the end-user will see.

4. Important Details:
    - `value` Attribute: This is used for internal identification and should match the `subcategory` attribute in the place details.
    - `text` Attribute: This is what users will see on the website.

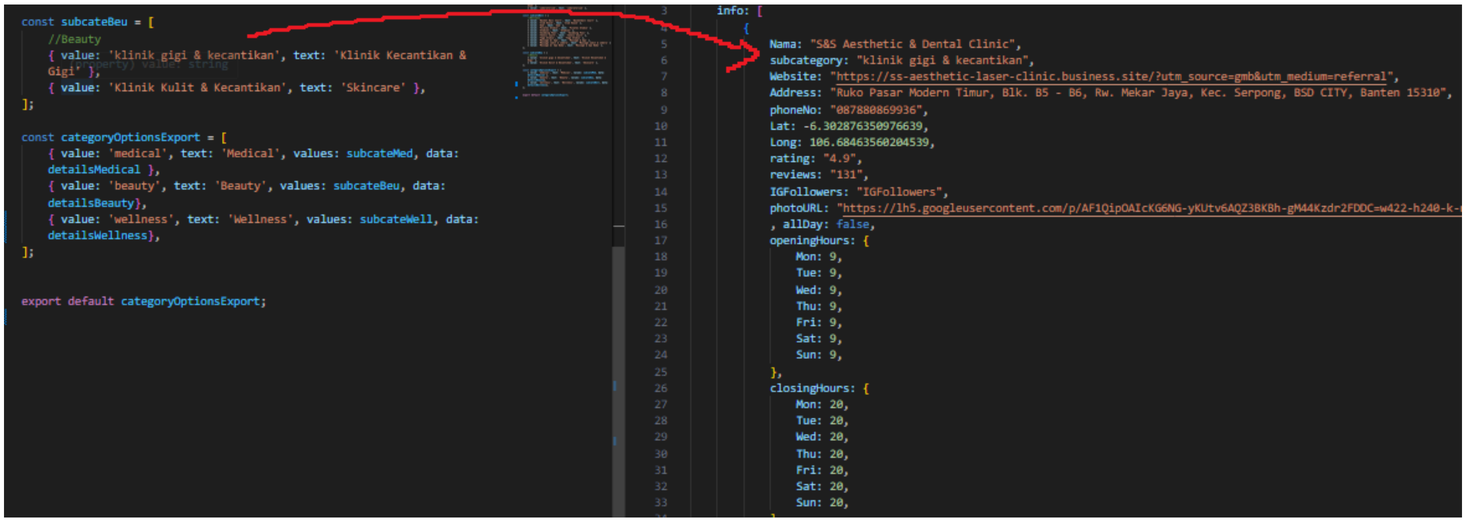
- Case Sensitivity: **Remember that the system is case-sensitive**. Ensure that the `value` for the subcategory matches exactly when referencing it in the place details, including capitalization and spaces.

Refer to Image VIII to see how a subcategory is used in a place's details.



```
const subcateBeu = [
    //Beauty
    { value: 'klinik gigi & kecantikan', text: 'Klinik Kecantikan & Gigi' },
    { value: 'Klinik Kulit & Kecantikan', text: 'Skincare' },
];
```

Image VII. An example of an existing subcategory

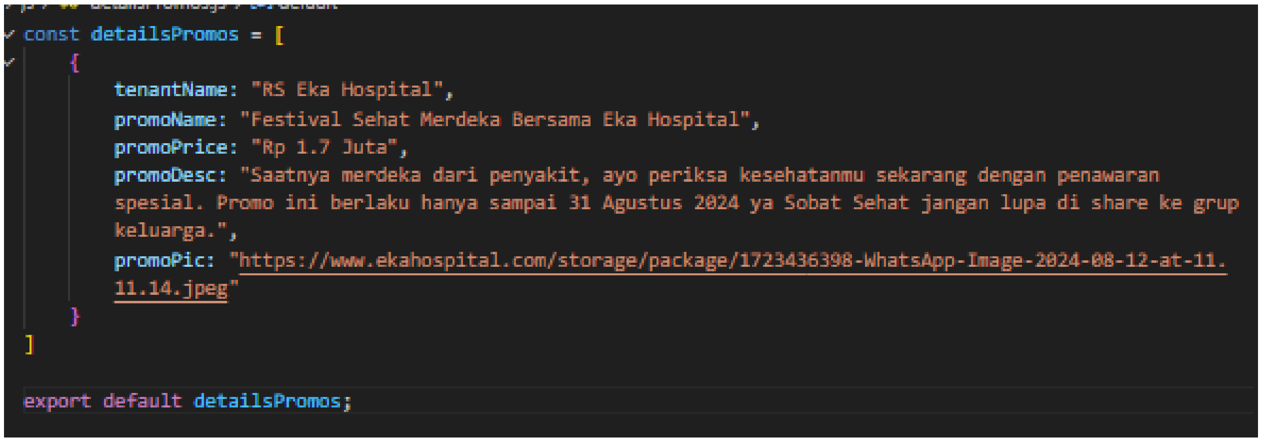


```
const subcateBeu = [
    //Beauty
    { value: 'klinik gigi & kecantikan', text: 'Klinik Kecantikan &
    Gigi' },
    { value: 'Klinik Kulit & Kecantikan', text: 'Skincare' },
];

const categoryOptionsExport = [
    { value: 'medical', text: 'Medical', values: subcateMed, data:
    detailsMedical },
    { value: 'beauty', text: 'Beauty', values: subcateBeu, data:
    detailsBeauty},
    { value: 'wellness', text: 'Wellness', values: subcateWell, data:
    detailsWellness},
];


export default categoryOptionsExport;
```

```
    info: [
        {
            Nama: "S&S Aesthetic & Dental Clinic",
            subcategory: "klinik gigi & kecantikan",
            Website: "https://ss-aesthetic-laser-clinic.business.site/?utm_source=gmb&utm_medium=referral",
            Address: "Ruko Pasar Modern Timur, Blk. B5 - B6, Rw. Mekar Jaya, Kec. Serpong, BSD CITY, Banten 15310",
            phoneNo: "087880869936",
            Lat: -6.302876350976639,
            Long: 106.68463560204539,
            rating: "4.9",
            reviews: "131",
            IGFollowers: "IGFollowers",
            photoURL: "https://lh5.googleusercontent.com/p/AF1QipOAIcKG6NG-yKUtv6AQZ3BKBh-gM44Kzdr2FDDC=w422-h240-k-
            , allDay: false,
            openingHours: {
                Mon: 9,
                Tue: 9,
                Wed: 9,
                Thu: 9,
                Fri: 9,
                Sat: 9,
                Sun: 9,
            },
            closingHours: {
                Mon: 20,
                Tue: 20,
                Wed: 20,
                Thu: 20,
                Fri: 20,
                Sat: 20,
                Sun: 20,
```

Image VIII. Example of Subcategory being used in a Place's Subcategory.

V. Adding Promos



```
const detailsPromos = [
    {
        tenantName: "RS Eka Hospital",
        promoName: "Festival Sehat Merdeka Bersama Eka Hospital",
        promoPrice: "Rp 1.7 Juta",
        promoDesc: "Saatnya merdeka dari penyakit, ayo periksa kesehatanmu sekarang dengan penawaran
        spesial. Promo ini berlaku hanya sampai 31 Agustus 2024 ya Sobat Sehat jangan lupa di share ke grup
        keluarga.",
        promoPic: "https://www.ekahospital.com/storage/package/1723436398-WhatsApp-Image-2024-08-12-at-11.
        11.14.jpeg"
    }
]

export default detailsPromos;
```

Image IX. *detailsPromos.js*

Promos are specified in the `detailsPromos.js` file. Each promo has five attributes:

1. Attributes to Include:
   - tenantName: The name of the tenant or place offering the promo.

   - promoName: The name of the promotion.

   - promoPrice: The price associated with the promotion.

   - promoDesc: A description of the promotion.

   - promoPic: A link to a photo for the promotion (optional).

2. Mandatory Attributes:
   - tenantName, promoName, promoPrice, and promoDesc are required for every promo. These attributes are straightforward:

   - tenantName: Must exactly match the name specified in the system (e.g., "Eka Hospital" should be spelled exactly as it appears in `detailsMedical.js` to ensure the system links the promo to the correct place).

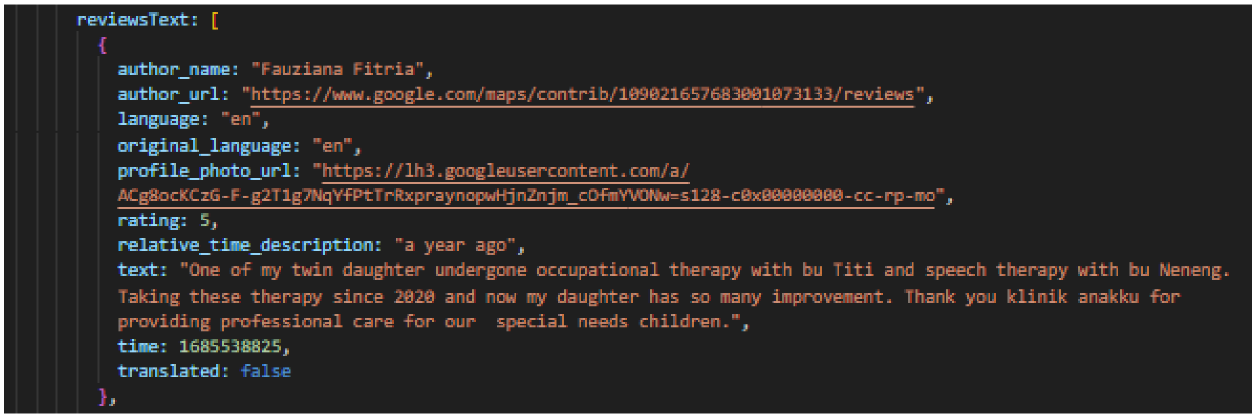
   - promoName, promoPrice, and promoDesc: These should clearly describe the promotion.

3. Optional Attribute:
   - promoPic: Works like `PhotoURL` in Section II. It should be a link to an image file (must end in .png, .jpg, or .jpeg). This attribute is optional.

4. Important Note:
    - Only one promo can be attached to each location at a time. After a promo has ended, it should be removed from the system.

VI. Reviews

I. Manual Insertion



```
reviewsText: [
  {
    author_name: "Fauziana Fitria",
    author_url: "https://www.google.com/maps/contrib/109021657683001073133/reviews",
    language: "en",
    original_language: "en",
    profile_photo_url: "https://lh3.googleusercontent.com/a/
    ACg8ocKCzG-F-g2T1g7NqYfPtTrRxpraynopwHjnZnjm_cOfmYVONw=s128-c0x00000000-cc-rp-mo",
    rating: 5,
    relative_time_description: "a year ago",
    text: "One of my twin daughter undergone occupational therapy with bu Titi and speech therapy with bu Neneng.
    Taking these therapy since 2020 and now my daughter has so many improvement. Thank you klinik anakku for
    providing professional care for our  special needs children.",
    time: 1685538825,
    translated: false
  },
```

Image X. *An example of a review*

Reviews are optional but recommended attributes that you can add to each place, as mentioned in Section II.

1. Adding Reviews:
    - Reviews are added under the `reviewsText` attribute for each place.

    - There is no limit to the number of reviews you can add, but it’s recommended to include up to five reviews per place.

2. Basic Format for Reviews:

    - Here is the simplest format for a review:

```
{
 author_name: "Example Author",
 rating: 5,
 text: "The place was good!\n\n Very good service!",
},
```

- author_name: The name of the person who wrote the review.

- rating: The rating given to the place (e.g., 5 out of 5).

- text: The content of the review. This can include feedback or comments about the place.

3. If No Review Text:
- If there is no text for the review, you can leave the `text` attribute as an empty string:

```
{
  author_name: "Example Author",
  rating: 5,
  text: " ",
},
```

II. Automated Insertion

Due to the limitations and the limited time provided to implement this program, there is no fully automated way to provide updates to the review mechanism. However, it is partly automated but still requires a person to manually activate it.

- The updater file can be found in the root folder of the project, under the folder of ‘websiteMaintainence’ and named as update.js.

- This file utilizes Google Maps API to collect the placeIDs of the location names before querying for the reviews attached to their locations.

- However, due to technical difficulties, not all locations are guaranteed to be found.

A guide to utilizing update.js is as follows. If it is too convoluted, please consult your IT staff regarding ‘running a file with node.js’, as they would have the technical expertise to do so.

The guide is as follows,

Step 1: Install Node.js

For Windows Users

1. Download the Installer:
- Go to the [Node.js official website](https://nodejs.org/).

- You will see two versions available: LTS (Long Term Support) and Current. For most users, the LTS version is recommended.

- Click on the LTS version to download the Windows installer (.msi file).

2. Run the Installer:
    - Open the downloaded `.msi` file.
    - Follow the prompts in the Node.js Setup Wizard.
    - Accept the license agreement and select the destination folder.
    - Choose the default settings unless you have a specific reason to change them.
    - Click "Install" and wait for the installation to complete.

3. Verify Installation:
    - Open Command Prompt (Press `Win + R`, type `cmd`, and press Enter).
    - Type `node -v` and press Enter. You should see the installed version of Node.js.
    - Similarly, type `npm -v` to check the installed version of npm (Node Package Manager).

For Apple Users:

1. Download the Installer:
    - Go to the [Node.js official website](https://nodejs.org/).
    - Download the LTS version for macOS.

2. Run the Installer:
    - Open the downloaded `.pkg` file.
    - Follow the instructions in the installer.
    - Accept the license agreement and proceed with the installation.

3. Verify Installation:
    - Open Terminal (you can find it in Applications > Utilities).
    - Type `node -v` and press Enter to check the installed Node.js version.
    - Type `npm -v` to check the installed npm version.

Step 2: Run Update.js

1. Open Terminal/Command Prompt:
    - Navigate to the directory where ‘websiteMaintainence’ is.

    - You can use the `cd` command to change directories. For example:

```bash
cd path/to/your/file
```

    - Replace `path/to/your/file` with the actual path where ‘websiteMaintainence’ is located on your device.

2. Run the File with Node.js:
    - Type `node update.js` and press Enter.

Once you have completed step 2 of step 2, flurry of lines of text will show up on screen. Do not worry, this means the system is working as intended. At the end, you will see the message of ‘Finished Updating Places’.

Of course, due to the limitations of the system, several locations may still lack reviews, as such I recommend giving the list a combing through to find what places lack reviews and follow the steps in Section VI.I ‘Manual Insertion’ in how to add them manually.

VII. Contact

If things go wrong, please contact your IT Department or those in charge of maintaining this project in order to resolve the issues.

The original creator of the system can be contacted via jasonalexanderyuwono@gmail.com