Overview

You have been tasked with adding some new functionality to the Proctors Developer Store. Currently the site just has general information on it but the client would like a store finder.

We have provided you with a Drupal installation, a 'Store' content type, a custom module 'store_locator' and an example CSV file containing the stores.

The user admin account is **superuser** and the password is **superuser**. You must be logged in in order to use the debug function dpm() and dvm() (see the comments in the code).

Task 1 - CSV Store Importer

You will be given a CSV containing the list of stores and will need to write an importer that adds the data to a database table.

The custom module (***sites/all/modules/custom/store_importer***) has a menu callback already defined which you should use: store_locator/import. Simply add your import code to the store_locator_import_submit() function. The data from the form will be in the variable $form_state['values']['csv_data'].

For simplicity, you don't have to deal with file uploads. The user will just paste the CSV content in to the box.

Some of the things we are looking for are:

- Data being imported in to a DB row correctly.
- Checking if a store already exists in the DB.
- Any performance considerations.

There is a CSV with sample data to download in the instructions folder (/instructions/store_data.csv) or click here.

Task 1 - Extension

The client doesn't know the Latitude and Longitude of all their stores. Use the Google maps geocoding API to grab them as you import the data.

https://developers.google.com/maps/documentation/geocoding/intro

You should be able to grab the details from Google in a few lines of code.

You shouldn't need an API key for this. But here is one you can use if you really need to:

AIzaSyDkn1amsOkmbqBDQl0uJlorDNfRYbnk2u8

Task 2 - Store List Page

The client wants users who visit the site to be able to find the stores. We have defined another menu hook at store_locator/list. Insert your logic into the store_locator_view_all() function. This function should return HTML, which will be rendered on the page.

We're looking for a list of stores to show:

- Store name.
- Store type.
- A link to view the full details.

A bonus would be if the user could filter the list by store name and store type.

Use some custom CSS (You will find a blank CSS file in the store_locator module 'sites/all/modules/custom/store_locator/store_locator.css') to arrange the elements on the page.

Task 2 - Extension

The client wants the stores of type 'Superstore' to appear at the top of the list by default.

Modify the list to achieve this.

Task 3 - Store Page

The designer has provided you with a "design". Your job is to make the store page match the layout of the design below.

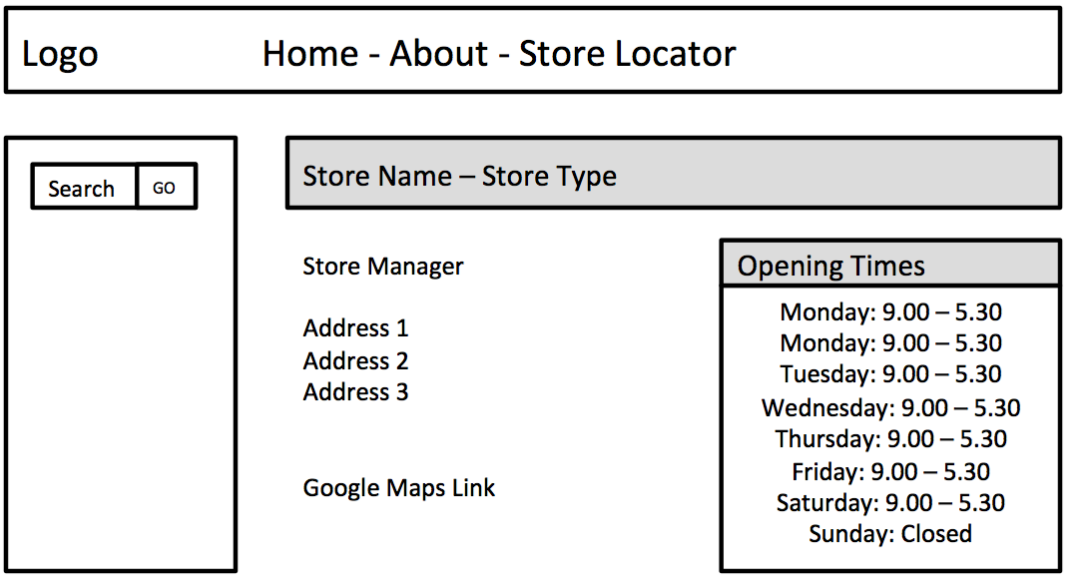
We have defined another menu hook for individual store pages. Create a store detail page which responds to this hook using the function store_locator_view_store() to output HTML.

Some of the things we are looking for are:

- Correct retrieval of the data.
- HTML tag choice.

- Good data access practices. (Within reason!)

You will find the full wireframe in the folder 'instructions' if you need it.



Task 3 – Extension

Insert a link to Google maps that shows the location of the store.

Tips for using a vagrant/ansible box

- You have to wait a second or two for everything to sync to the virtual machine. Your code may not refresh immediately.
- Drush is available. you can run commands on the virtual machine using the alias @vagrant e.g. drush @vagrant cc all
- If you make changes on the virtual machine (e.g. drush @vagrant dl devel) they will be overwritten next time it syncs. Make all changes to code locally (e.g. drush dl devel).