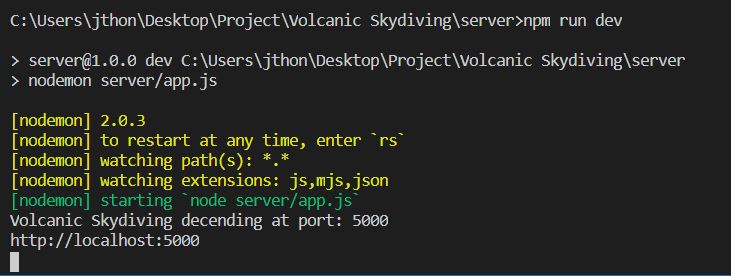
User Guide:

Open the server folder and open a terminal

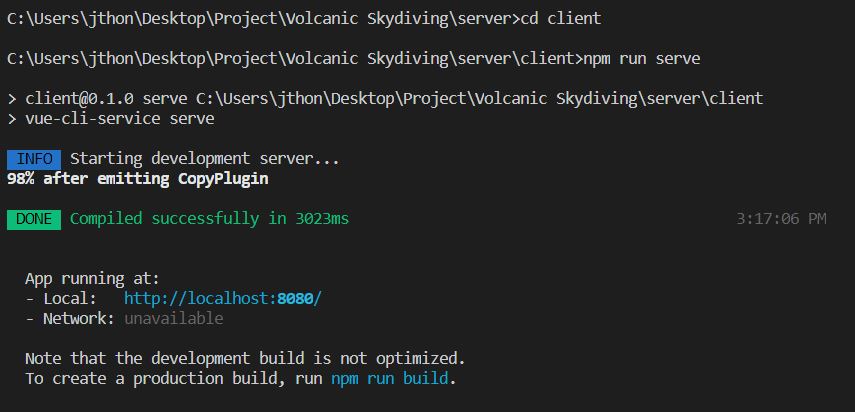
You start the service by typing “npm start dev”



The server runs on <http://localhost:5000> (port 5000)

Open another terminal and type “cd client” to switch to the client folder.

You start the client service by typing ‘npm run serve’



The client side runs on <http://localhost:8080> (port 8080)

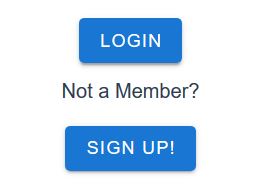
The database name is ‘volcano’ consisting of 2 collections.

The 2 collection names are ‘volcanos’, and ‘members’

\*I provided csv files along with Json formats to import the data.

Upon navigating to <http://localhost:8080>, the user will come across the welcome screen.

If the user isn’t registered yet, they click the ‘sign up!’ button on the welcome page.



Alternatively, once the user has already made an account, all they need to do is click the ‘Login’ button.

Once the user has registered and signed in, they are redirected to the home page.

The homepage consists of volcano destinations listings providing an overview of each volcano.

To add a volcano, click on ‘Click here to add volcano’ link on the top

Fill out all the fields required and click the ‘Submit’ button.

To clear the form, click the ‘Clear’ button

To access more info about a particular volcano, click on the select button for the respective volcano.

Clicking the select button will direct you to the volcano details component.

The volcano details allow the user to make changes, either by updating a field or deleting the document.

The user simply clicks the begin edit button on the bottom of the screen and three new buttons show up

To update a record: click the ‘Update’ button

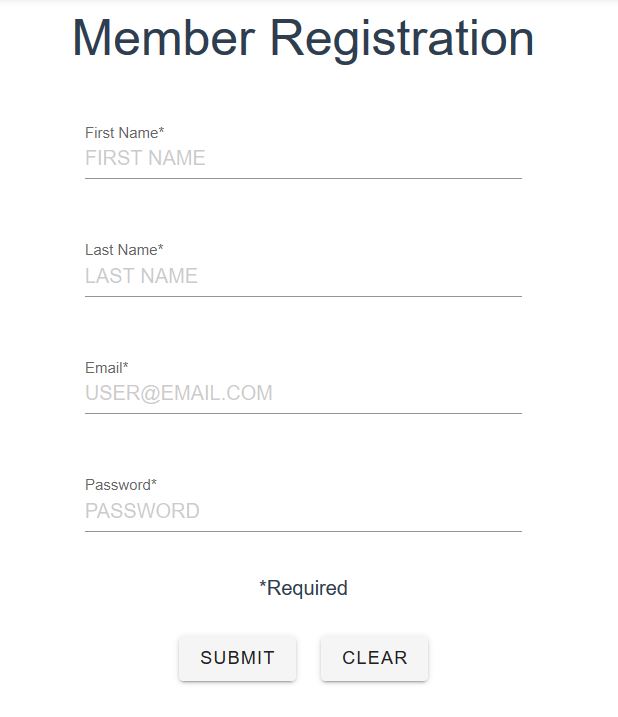
To delete a record: click the ‘Delete’ button.

To return home and see the changes, click return home button.

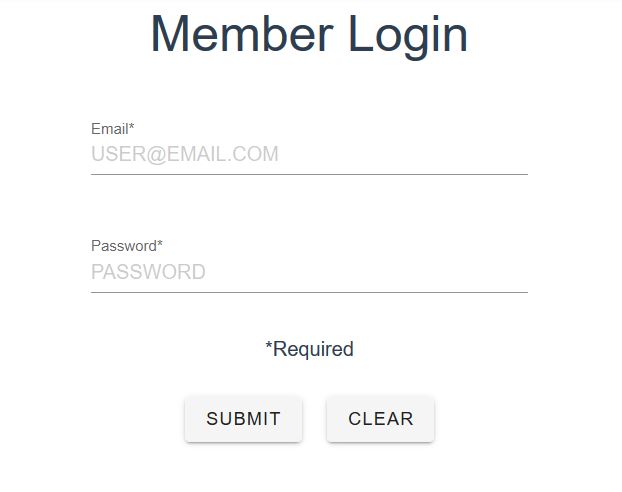
\*More information and screenshots are shown below.

The components my application uses are:

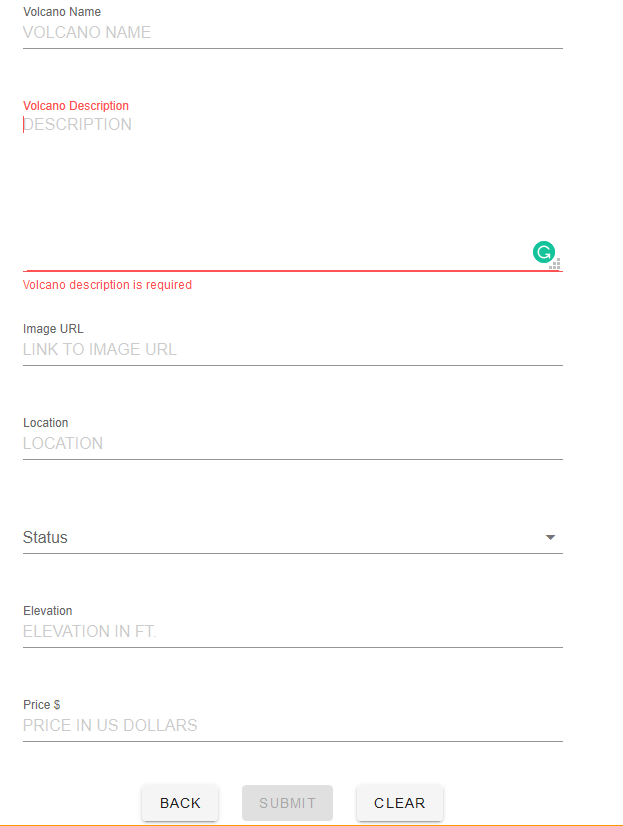
**Register.vue:** The register view allows for members to sign up before accessing the homepage that takes them to the volcano destination listings. I created a form that takes the first name, last name, email, and password parameters and if validations are met, it will redirect you to the login screen.



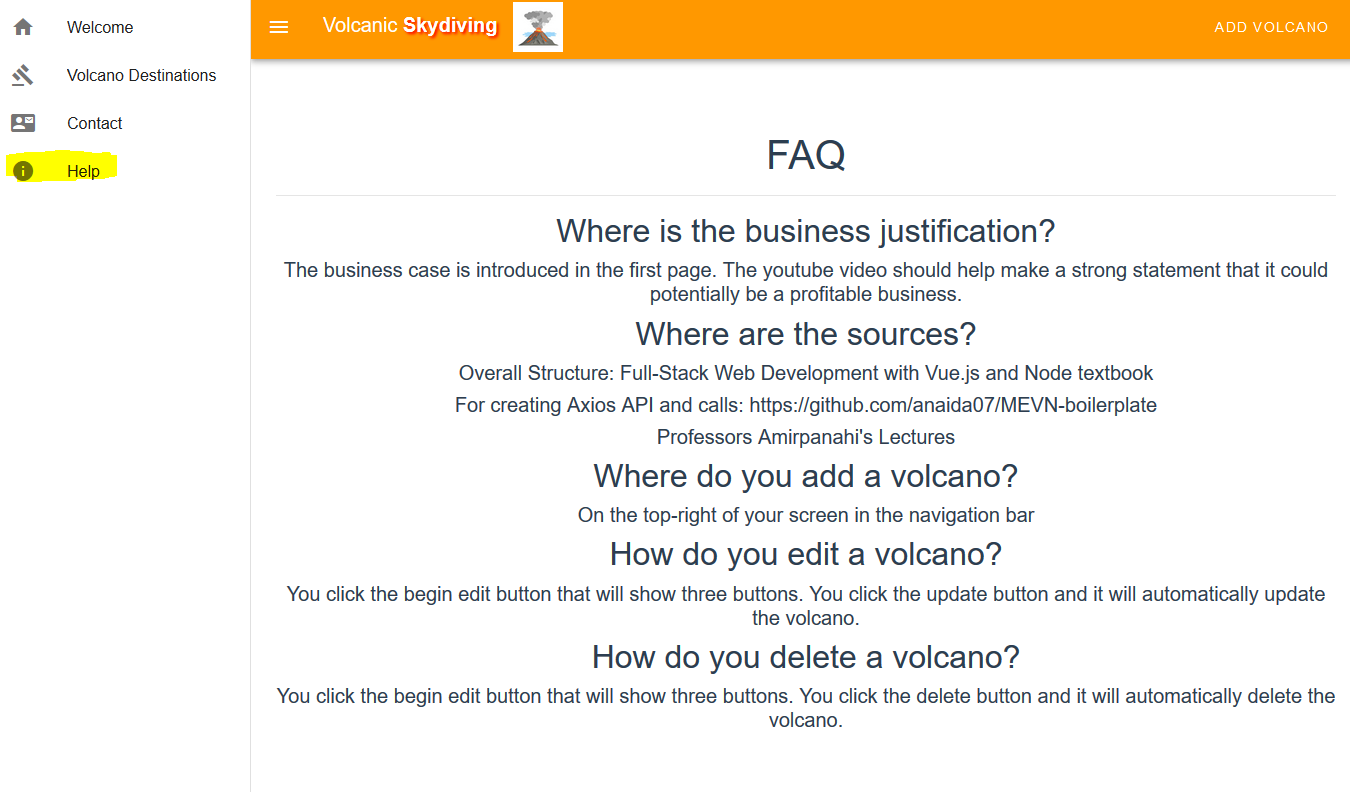
**Login.vue:** This view allows users to log in. I created a form that takes the email and password parameters and upon validation, redirects them to the homepage. (Which is the Volcano component)



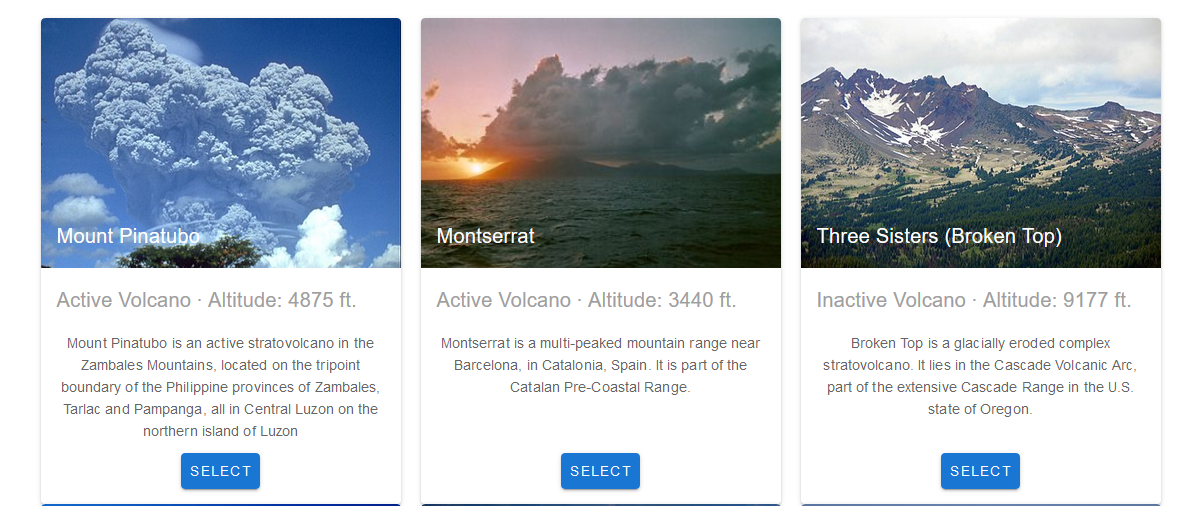
**AddVolcano.vue:** This component allows a user to add a new volcano destination by providing the fields required. It also adds validation by making sure all fields are filled out or the user can’t submit the form.



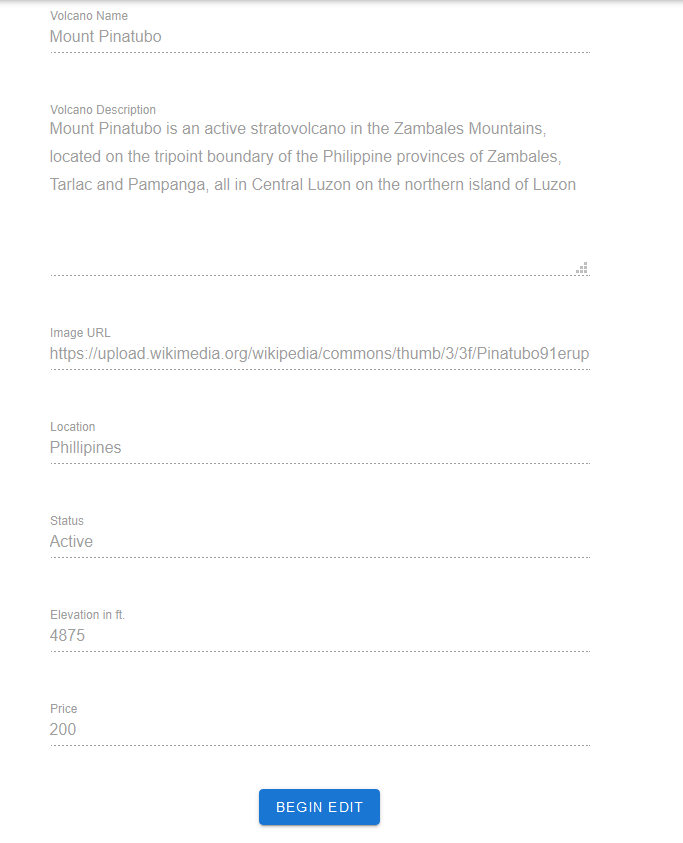
**Help.vue:** This component answers questions that users may have. It is accessed by the left navigation pane.



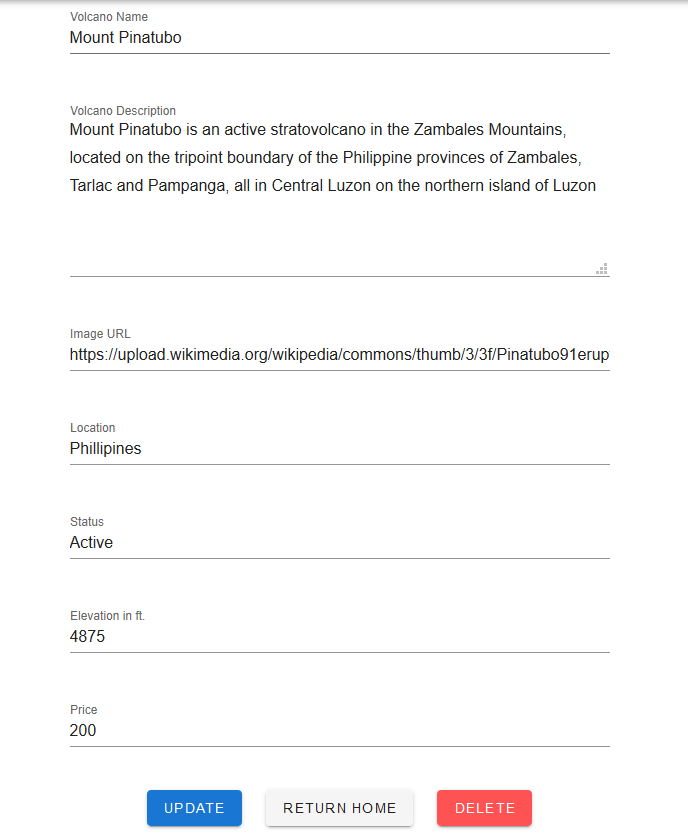
**Volcano.vue:** This is the main page for displaying all of the volcanos in the collection. When the user clicks on the ‘Select’ button, they are redirected to the volcano details page where they can view more information.



**VolcanoDetails.vue:** This component allows the user to view information about the volcano and if they spot a mistake or simply wants to remove the volcano from the database, the user clicks the begin edit button on the bottom of the screen and 3 new buttons will display. Updating the volcano will update the volcano in the database, and the user can click the return home button to verify the correct changes. Deleting the volcano will prompt the user for confirmation, and upon confirming, the volcano is deleted and taken back to the home page.



After clicking “Begin Edit” button.



**Welcome.vue:** This displays when the user first accesses the website. I try to make a strong business case that will hopefully lure in customers. The user can login or register by clicking one of the buttons.

**App.vue:** Displays the overall structure of the application. I use Vuetify to attract users to navigate through the application. The navigation drawer (left side), the app bar (top), and the footer (bottom) stay consistent throughout the whole application.

