2024

CAB230 Assignment 2 Client Side

CAB230

Volcano API – Client Side Application

Shafi Uddin

N11245409

5/10/2024

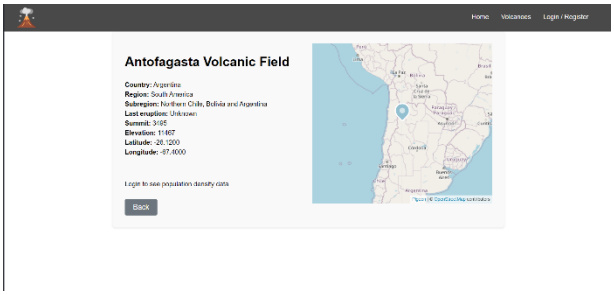
Contents

Introduction

Purpose & description

The purpose of this application is to provide an interactable react application that allows users to access comprehensive data relating to the world's volcanoes. Users can search for volcanoes, view details such as their location via an interactive map, height of their summit, as well as when they last erupted. Users are also able to create accounts to log into the site and access exclusive content such as population density data.

The application implements all the prescribed features, such as the ability to login and register, search for volcanoes and accessing exclusive content using an account. An interactive map and table displaying population density data are also present. The pages are presented in simple straightforward layouts to make navigation easy, and high contrast elements are used throughout. Users are also presented with error messages or alternative messages if errors are encountered or if it is taking longer than usual to load data.



Completeness and Limitations

All prescribed features have been implemented and do not have any persistent bugs. The prototype meets a standard of 6 or 7 since:

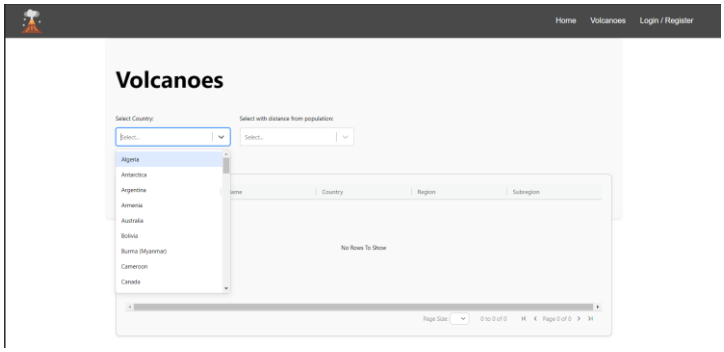
- All endpoints are implemented
- Exclusive population density data is only accessible when user is signed in
- Table component is implemented with pagination to show fetched volcanoes from user search query with the ability to filter and sort results
- Excessive numbers of queries are not sent to the server
- CSS styling used to provide an aesthetically pleasing design that is also simple to navigate
- An interactive map component as well as an interactive chart are also used to display certain data relating to volcanoes

The only persisting issue is the logout button, which does not render immediately after the user signs in, it only appears after the user navigates away from the login/register page.

Use of End Points

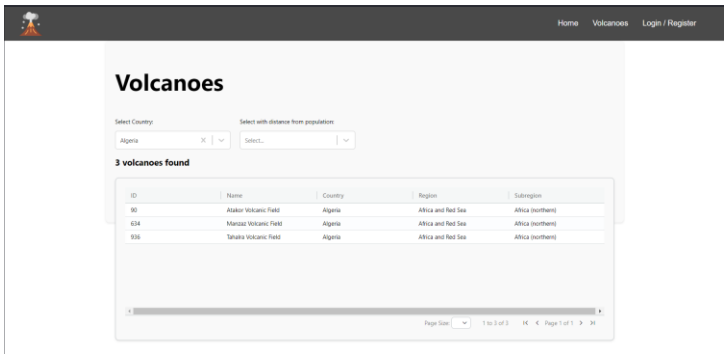
All end points of the external web-API have been utilized as shown below:

/countries

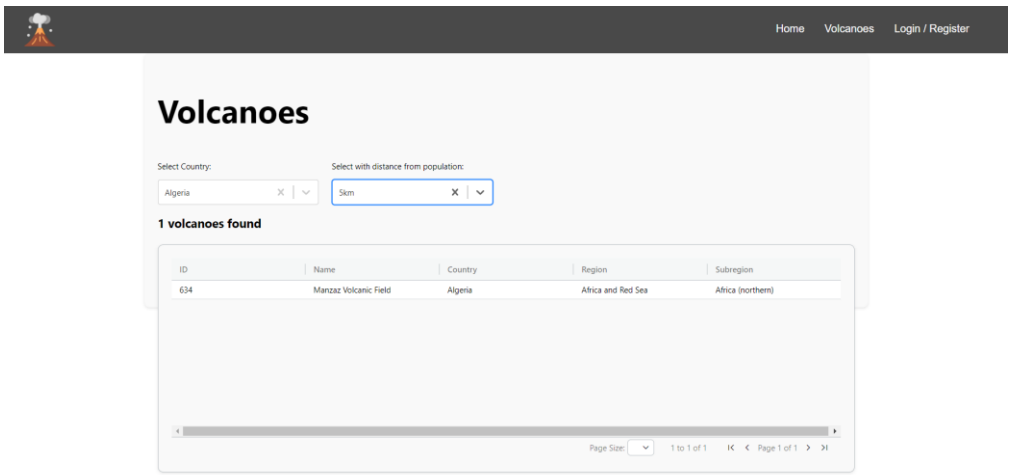


The countries are fetched using the /countries endpoint and placed in the left search bar. This allows the user to filter their search based on the country the volcano is located in.

/volcanoes

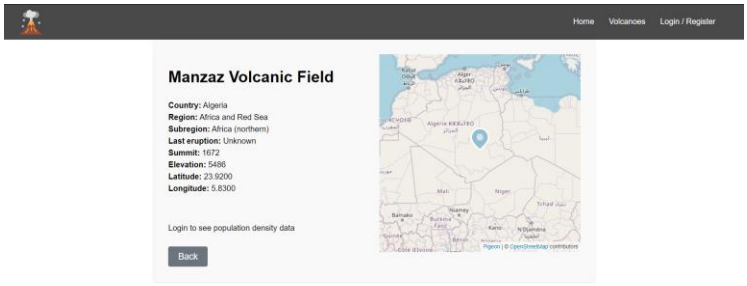


The volcanoes are fetched using a selected country as seen above.

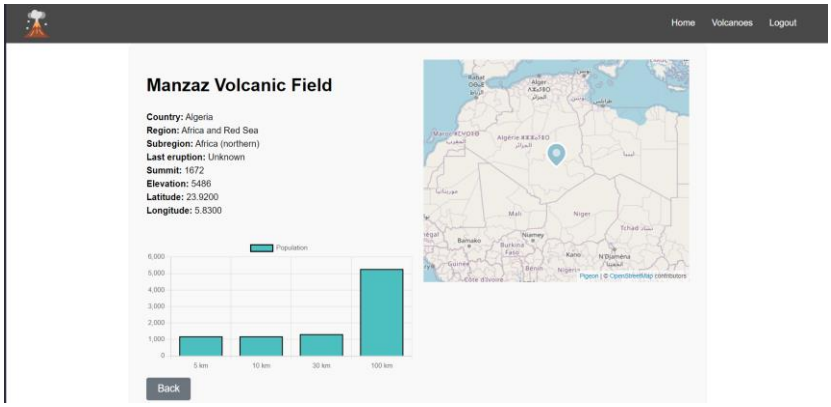


The search can also be further filtered by choosing volcanoes that have a population that is within a certain distance from the volcano.

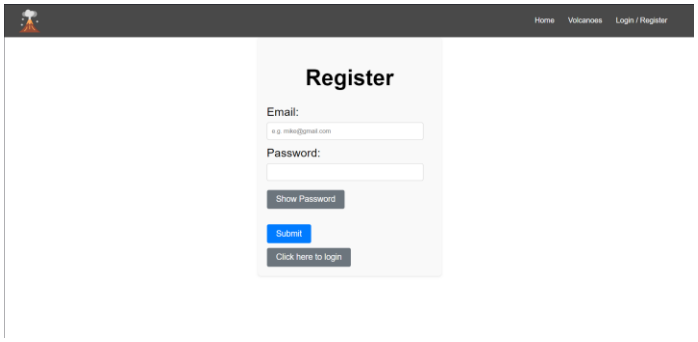
/volcano/{id}



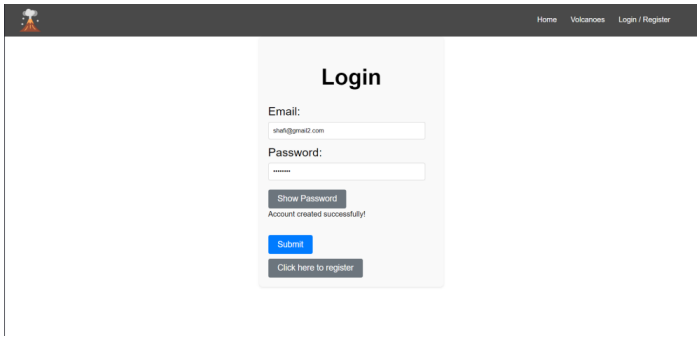
The endpoint is used to fetch data relating to the volcano and present it in text, on an interactive map as well as chart that can only be accessed by signing in as shown below:



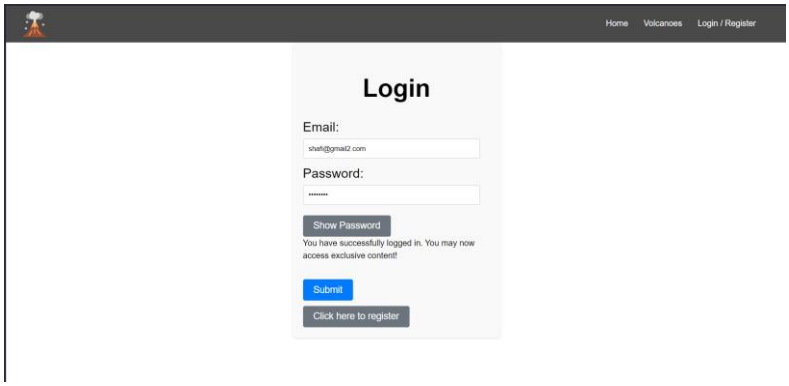
/user/register



Register page used to get new user data from user. However, by design, the application switches to the login page as soon as the account is created for the convenience of the user (as shown below).



/user/login



Message to show that the user has successfully logged in.

Modules Used

Several modules were used to create the web application:

Ag-grid-react

Module to provide fully-featured table components, including sorting and filtering. This module was used to display the search results of the volcanoes.

https://www.ag-grid.com/react-grid/

React-select

A module that allows for easy creation of functional and well designed search bars. Used to help the user filter volcanos by country and distance from populations.

https://react-select.com/home

Pigeon-maps

Used to pinpoint locations of volcanoes on an interactive map. User can zoom in and out and pan the map for a good view of the volcano.

https://pigeon-maps.js.org

React-chartjs-2

Used to create a chart that presents the population density near the volcano.

https://www.npmjs.com/package/react-chartjs-2

React-router-dom

An extension of React-router but with additional features that allows for the declaration of navigation paths, history management, and allows for efficient rendering of the page.

https://www.npmjs.com/package/react-router-dom

Application Design

Navigation and Layout

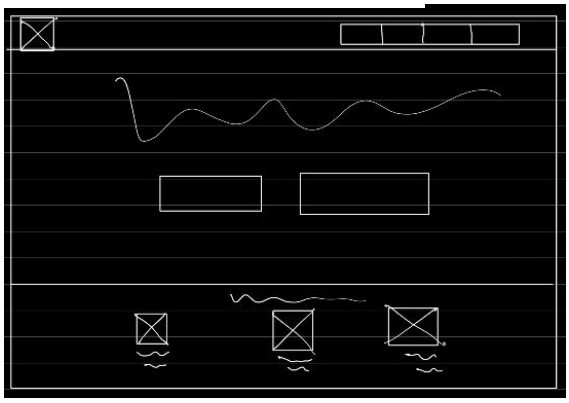
The navigation system was designed to be easy to use, efficient and reliable. All pages of the website (except for data about individual volcanoes) can be accessed from the top navigation bar. Other options considered included a navigation bar that is hidden unless the user presses a button in one of the top corners. This was scrapped as it could end up confusing the user, and it would result in a two click process to access different parts of the application.

The layout of the site is minimal, and the most important pages are designed so the user does not need to scroll to see all the content (minus the home page). Elements are presented in such a way that would allow the user to find what they are looking for in the places they expect them to be. For example, the data for individual volcanoes, including the population density chart, are presented in the same column. Right next to that column is the interactive map which can be easily accessed without having to scroll down.

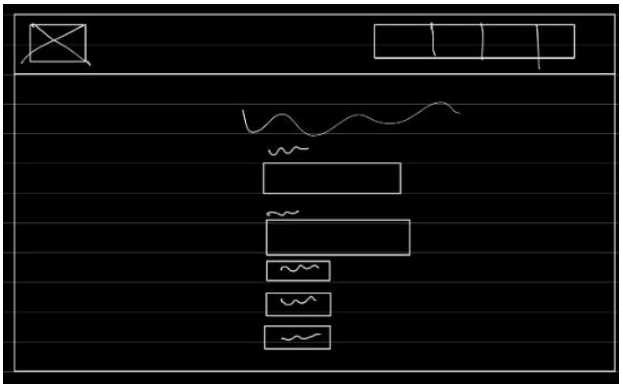
Design mock-ups

Mock-up wireframes were made to guide the design process of the web application's different pages. The overall structure and layout of the web application remains mostly the same, although minor improvements were made particularly to the layout of the navigation bar.

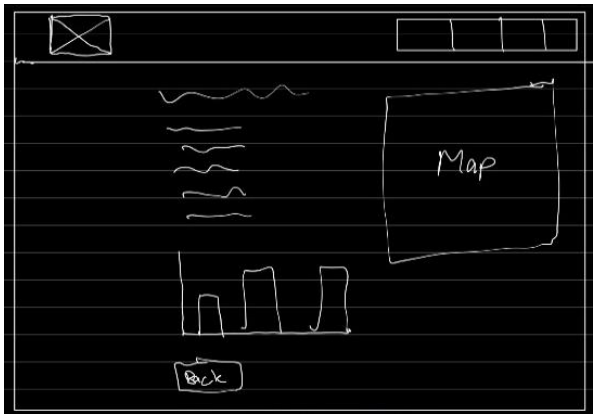
Home page



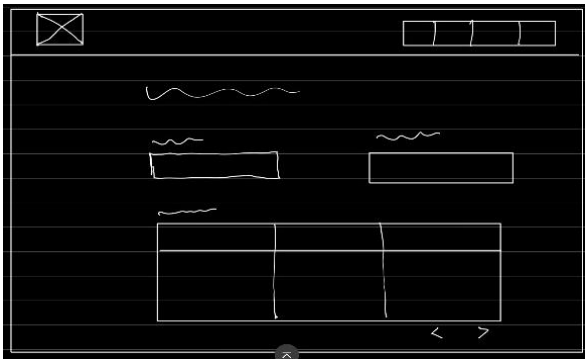
Login/Register page



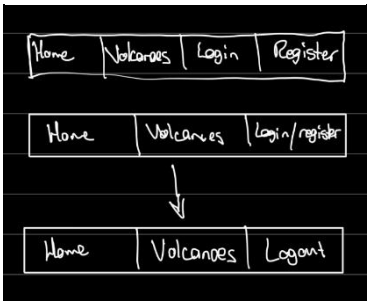
Volcano info page



Volcano search page

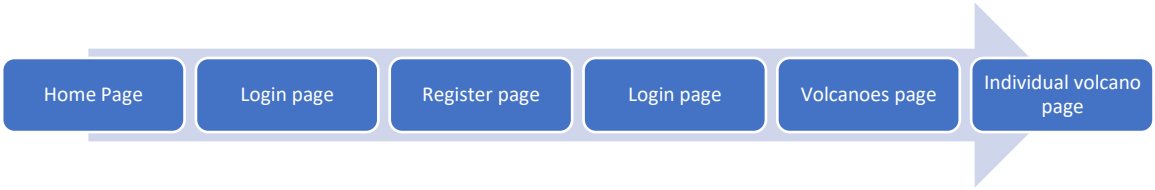


Evolution of navigation bar



Expected navigation path by users

The below flowchart showing the expected navigation sequence most users would do during their first usage of the app.



The login and register pages are technically the same page, the user toggles between registering or signing in using a button. The user is initially asked to sign in. Although inconvenient for new users, the trade-off is that they must press the toggle to create a new account before signing in, while returning users can log in without pressing more than one button.

Usability and Quality of Design

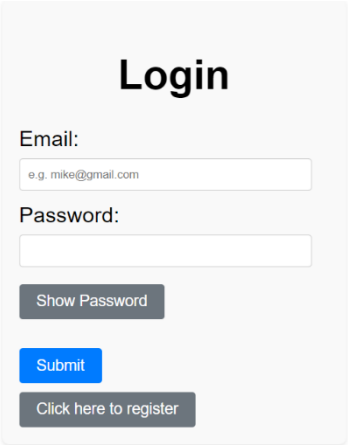
The layout of the web application is based on simplicity as well as being as efficient to use and view.

- The user can easily access the most important parts of the application usually with one or two clicks. All the most important parts of the application are accessible from the navigation bar which is always positioned at the top of the screen.
- The layout of the navigation bar as well as the search function conforms with the layout used in many websites such as W3Schools and The Weather Channel. These websites include a horizontal navigation bar which is easily accessible across most of the website.
- The design of the webpage is consistent since users would mostly work from top to bottom when trying to access or view data relating to volcanoes, or to sign in or register.
- Visually, the same colour scheme is used throughout. The exception is the home page which goes for a darker design compared to the rest of the application. The home page is designed to be as visually appealing as possible to welcome the user to the website.

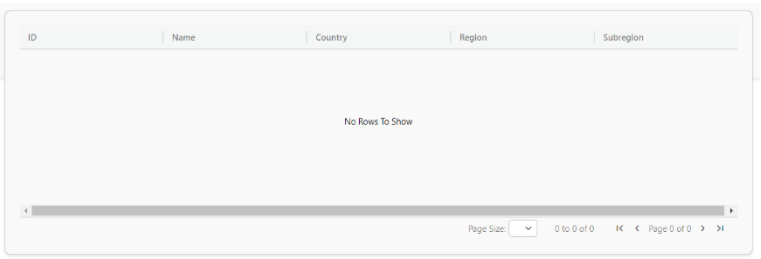
Accessibility

The web application was designed to be as accessible as possible. Measures taken to achieve this include high contrasting-coloured elements, ensuring that web contents are still viewable without CSS styling, minimising flickering, and clearly differentiated rows and column headers in tables.

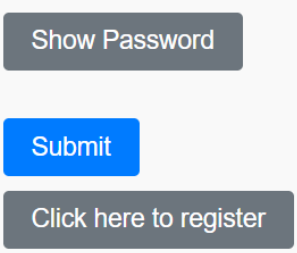
High contrast between all elements wherever possible. The website is predominantly white (#FFFFFF), off-white (#F9F9F9) and black (#000000) (see right). The pages are predominantly white, but the content is placed inside an off-white backdrop. The slight distinction between colours was made so that users can focus on the main content of the page. The contrast between the off-white backdrop and predominantly black elements are still significant. The contrast rating for the off-white backdrop and black text is 19.9:1 which meets WCAG testing guidelines.



The text present in tables (see below image) and search bars however is #707171, which results in a contrast rating of 4.6:1 which still passes WCAG requirements, although this could a point of improvement, especially since the text size is small.



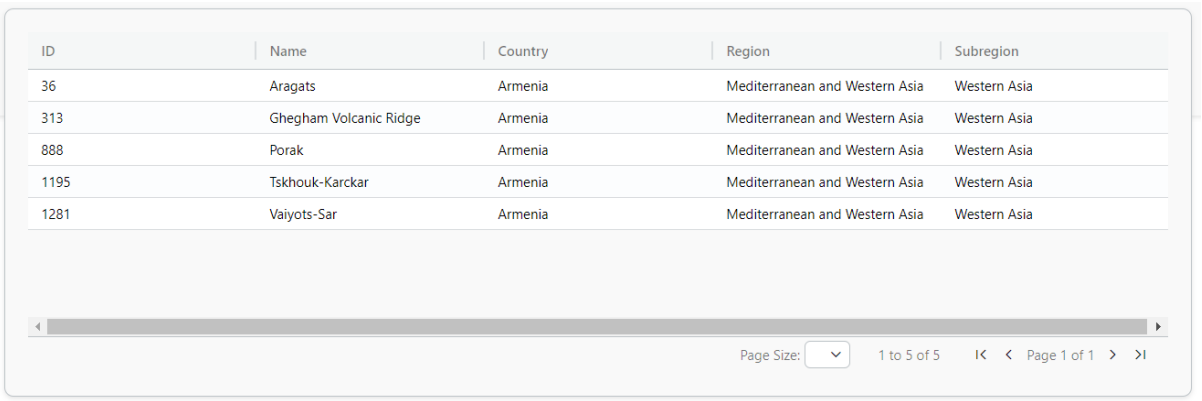
Some buttons (see right) are grey (#6C757D), or dark blue (#007BFF). These buttons don't contrast as well with the background. The contrast ratings of the two colours against the off-white background are 4.4:1, and 3.7:1 respectively. These colours still pass WCAG testing guidelines for large elements. The buttons use large text to compensate for the lack of contrast.



CSS styling is utilised heavily to create the layout of webpages. When CSS styling is disabled, some elements such as the table used to display search results does not display data properly, with some overlapping results. However, the web application is still mostly useable. This is a point of improvement.

Screen flickering is not present since dynamic content and animations are not used. This was done to minimize the risk of screen flickering, especially since it is not necessary for the web application in its current state.

The search results table has clearly defined rows and columns (see below image), to prevent confusion when users search for data.

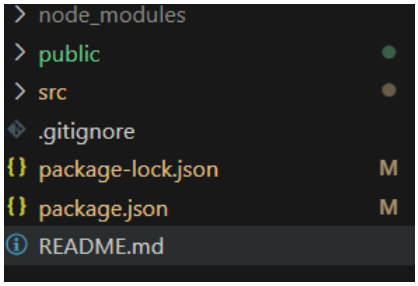


ID	Name	Country	Region	Subregion
36	Aragats	Armenia	Mediterranean and Western Asia	Western Asia
313	Ghegham Volcanic Ridge	Armenia	Mediterranean and Western Asia	Western Asia
888	Porak	Armenia	Mediterranean and Western Asia	Western Asia
1195	Tskhouk-Karckar	Armenia	Mediterranean and Western Asia	Western Asia
1281	Vaiyots-Sar	Armenia	Mediterranean and Western Asia	Western Asia

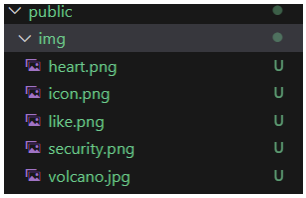
Technical Description

Architecture

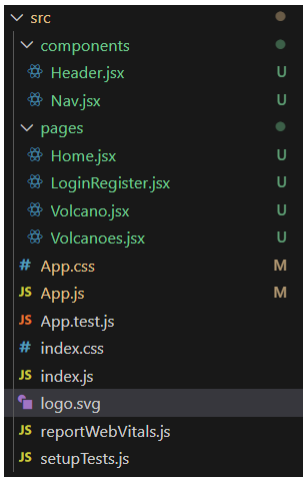
The react application uses the standard application architecture, containing a public folder and src folder (see right).



The public folder contains all the images used throughout the web application (see below).



The src folder contains all standard REACT files including App.js and App.css files. The application's pages are placed in a 'pages' folder, and reused elements such as the header and 'navbar' are placed in a components folder.



The header and navbar are accessed using a function call from the App.js file. The Nav function is called from within the header function

```
function App() {
  return (
    <BrowserRouter>
    <div className="App">
    <Header />
      {/* the content */}
```

```
export default function Header() {
  return (
    <header>
      {/* icon */}
      <div id="icon">
        <img src="img/icon.png" alt="Icon" />
      </div>

      <Nav />
    </header>
  );
}
```

Test plan

Manual testing was conducted to ensure the functionality of the web application in all expected, unexpected scenarios and where the application is expected to gracefully handle an error.

Task	Expected outcome	Result	Screenshot(s) (See appendix)
Register new user with valid email address	User is registered successfully	Pass	1
Register new user with existing details	User is told they already exist	Pass	2
Attempt register with only one field	User is told both fields need to be filled	Pass	3
Login with newly created user	User is told they have successfully logged in	Pass	4
Login with non-existent user (different from above)	User is told their username or password is incorrect	Pass	5
Attempt login with only one field	User is told both fields need to be filled	Pass	6
Search for volcano without distance filter	Volcano search results show	Pass	7
Search for volcano with distance filter	Volcano search results show	Pass	8
Click on search result without being logged in	All volcano details will show except for population data	Pass	9
Click on search result while being logged in	All volcano details will show including population data. They will be told to login to access said data.	Pass	10
User logs out from any page	User is logged out and taken to login page	Partial Fail	11

Attempt to login with incorrect URL for external API (testing to see if program handles server errors gracefully)	Application does not crash and error message is shown after failed to fetch request is caught.	Pass	12
Attempt to register with incorrect URL for external API (testing to see if program handles server errors gracefully)	Application does not crash and error message is shown after failed to fetch request is caught.	Pass	13
Attempt to fetch volcanoes from search with incorrect URL for external API (testing to see if program handles server errors gracefully)	Search bar for countries will be replaced with text saying the data is loading. App does not crash.	Pass	14
Attempt to view volcano with incorrect URL for external API (testing to see if program handles server errors gracefully)	Data will be replaced with text saying the data is loading. App does not crash.	Pass	15

Test case 11 is considered a partial fail, since the logout button does not render as soon as the user logs in as intended. This means they cannot log out as soon as they log in, and must navigate to some other part of application to log out (although this is an insignificant issue). They are still successfully logged out when they press the button when it is rendered. (See persistent errors section for more details.)

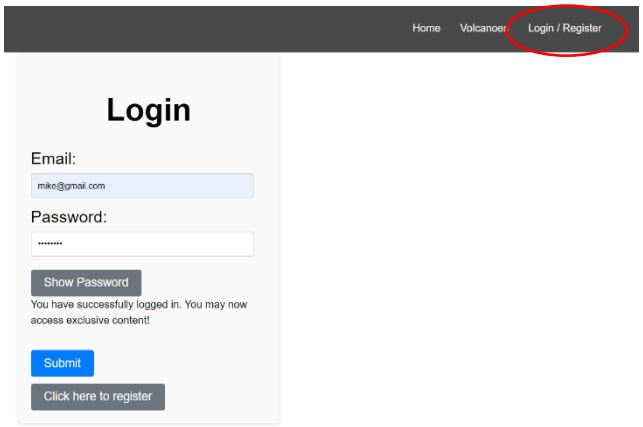
Difficulties / Exclusions / unresolved & persistent errors

Difficulties

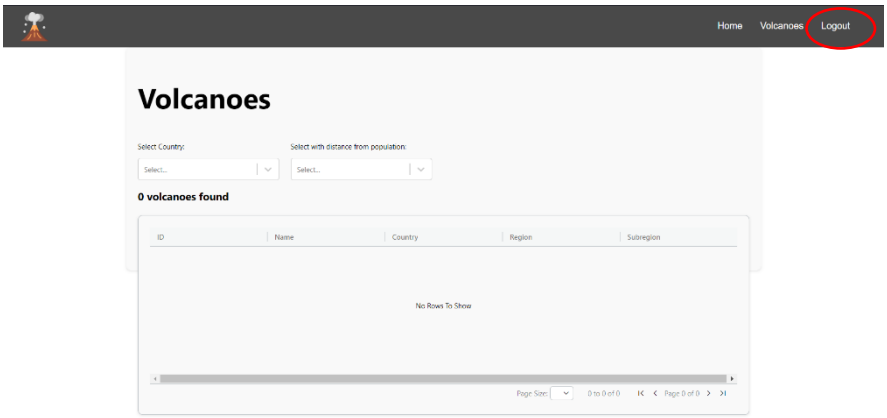
During development, there were a few roadblocks relating to the fetching of data from the web server and presenting them in the ag-grid table and the react-chart-js graph. This issue was resolved by experimenting with how data is handled in react (from fetching, to converting to JSON format, and then inputting it into the respective modules using function calls). All planned features, however, were successfully implemented.

Unresolved and persistent errors

As mentioned previously, the log out button does not render as soon as the user logs in. It only renders after they navigate to a new page.



In the image (left), the user has successfully logged in but the logout button has not appeared in the navigation bar (see red ellipse).



In the image (left), when the user has navigated to another page, the logout button does appear. It is here that the user can now logout.

The cause of this issue is due to the rendering system used by react, where not all components are re-rendered with changes made to the application due to performance optimisations. However, with the current implementation, the navigation bar is not updated when the user successfully signs in, only when the change pages.

In the current implementation of the program, when the user details are verified by the server and deemed to be correct, the following code is executed:

```
else {
  sessionStorage.setItem("token", res.token);
  setErrorSection(loggedIn);
}
```

In the navigation bar function, the following code is used to determine if the user is logged in and whether the logout button should be rendered:

```
useEffect(() => {
  //const token = sessionStorage.getItem("token");
  if (token !== null) {
    setIsLoggedIn(true);
  }
}, [token]);
```

```
<li>
  {isLoggedIn ? (
    <a href="/loginRegister" onClick={handleLogout}>Logout</a>
  ) : (
    <a href="/loginRegister">Login / Register</a>
  )}
</li>
```

The useEffect function is used to set the variable isLoggedIn to true when a change is made to the variable token, which is declared using:

```
const token = sessionStorage.getItem("token");
```

Although it was believed that the usage of useEffect would solve this issue by causing a re-render when the value of the token changes, this not the case. Solutions such as forcing the navigation bar to re-render were considered, but this is poor practice. Due to time constraints, and the fact this is an insignificant issue for a prototype, the issue was not rectified for the prototype build.

Extensions

The current web application could be expanded in terms of its feature set to make it more appealing to potential users. Future features include:

- Allowing academic papers on different volcanoes or regions to be published and accessed through the web application

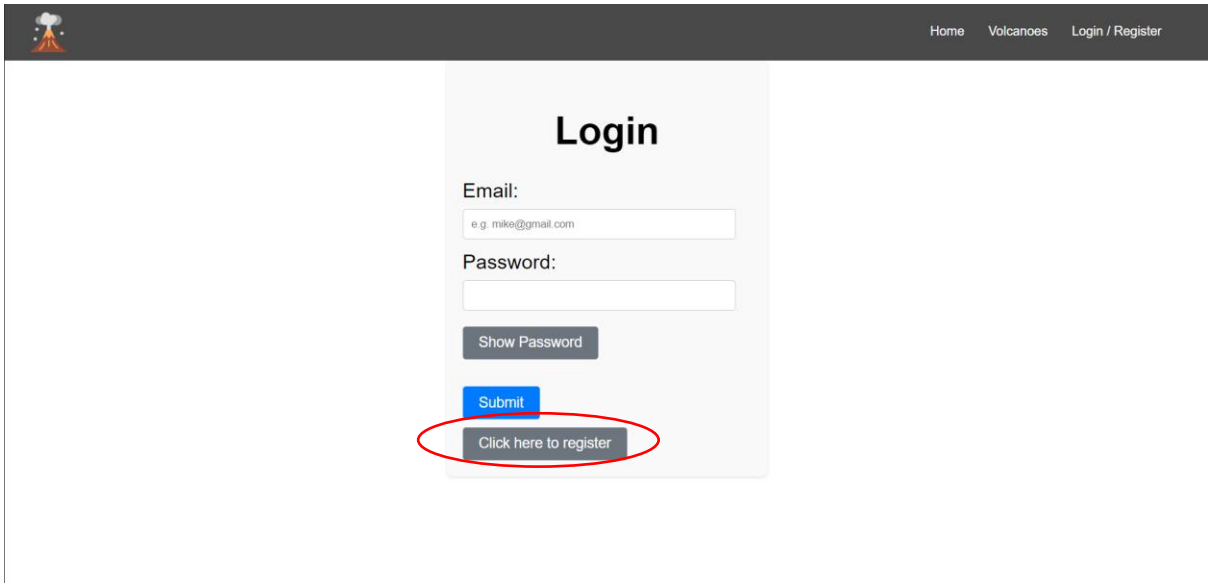
- Addition of a search bar where the user can look for individual volcanoes without first having to filter by country or distance from populations
- Showing the latest news stories relating to volcanic or tectonic activity
- Latest alerts relating to volcanic eruptions and safety information from authorities

User guide

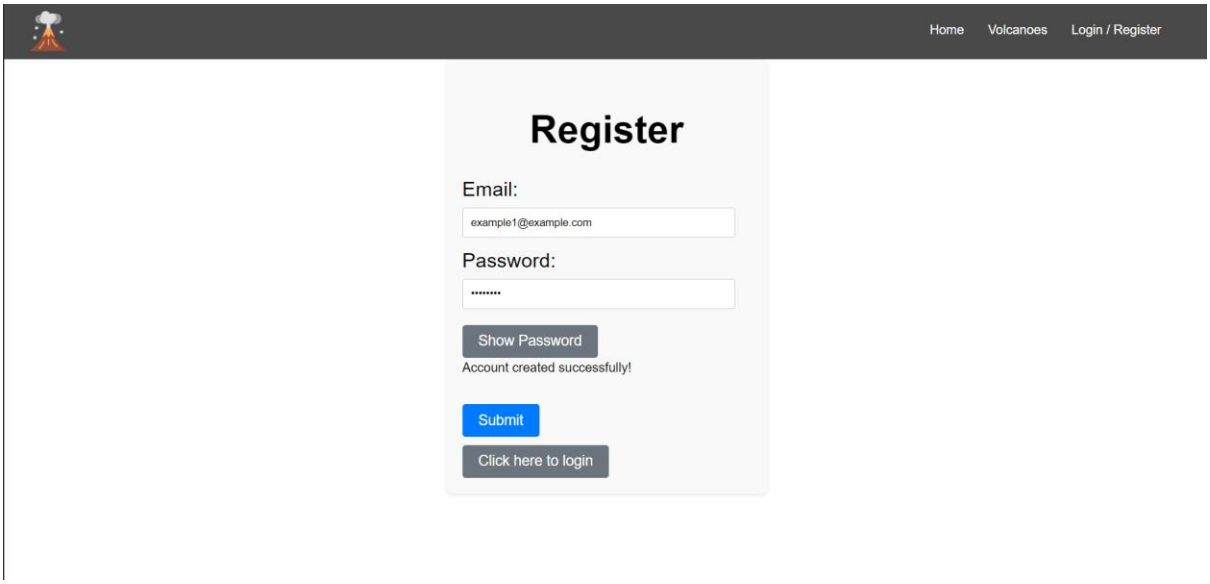
Below is a user guide to show how to access the data for a volcano:

1. Upon opening the web application you will start on the home page. Most users will likely press the Login / Register button either in the navigation bar on the home page itself.

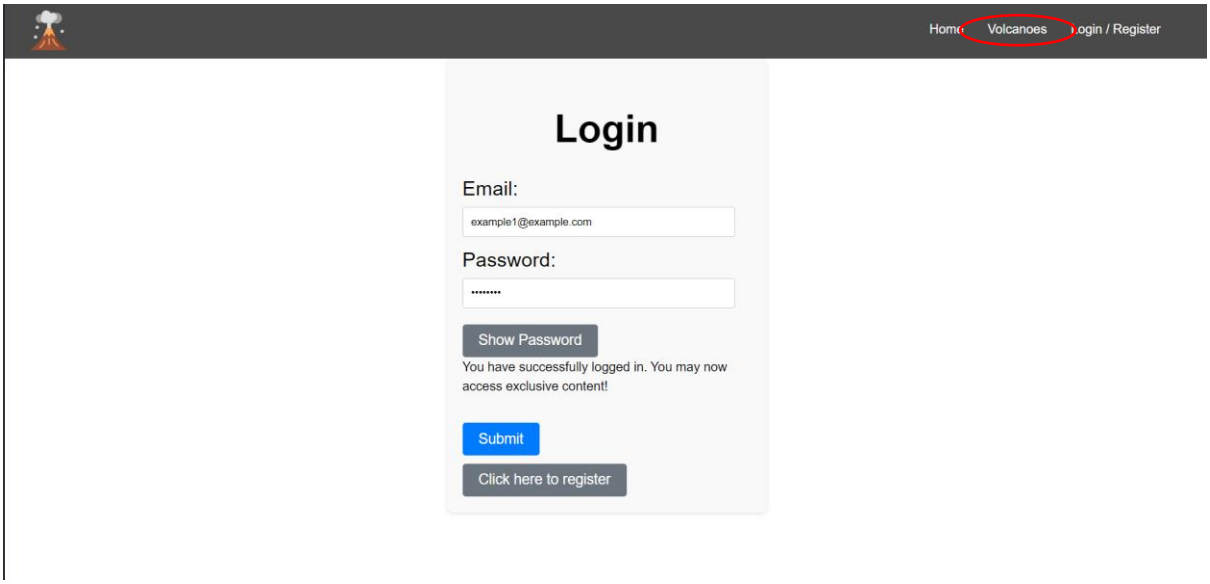
2. New users will either proceed to sign in or they will press the “Click here to register" button



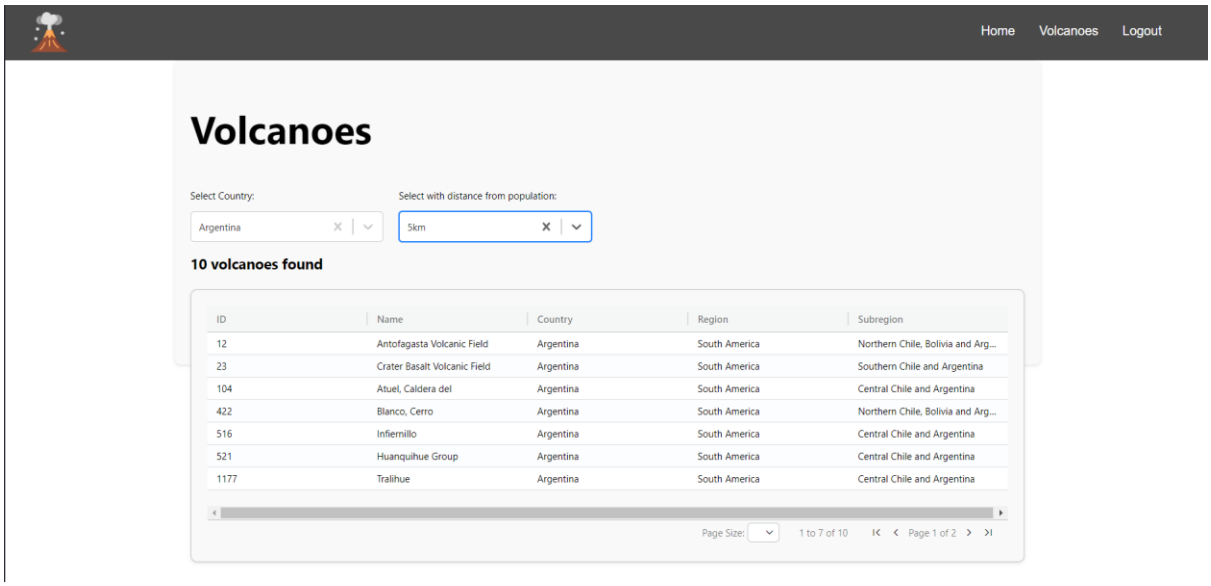
3. Users will enter their email address or password. The email will be validated to see if it is in the correct format. The format ensures that there is at least one character before the ‘@’ symbol, and at least one character after the ‘.’.



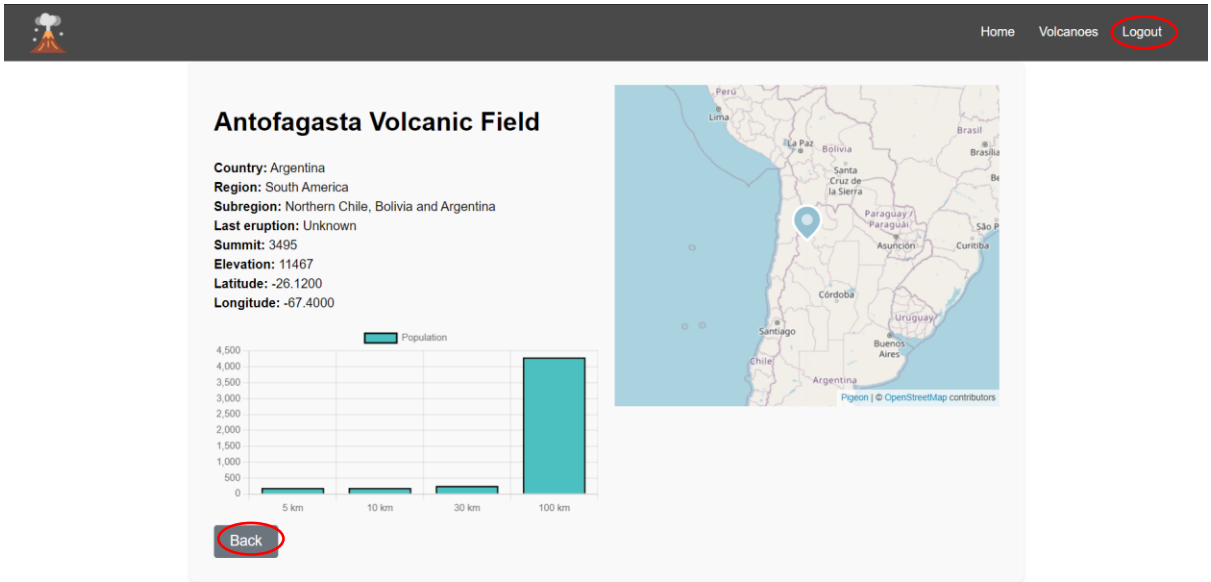
4. Users will then be directed to the login page automatically upon pressing submit. The fields will already be populated with their entered details, so they can simply press enter or submit without typing anything. They will then likely navigate to Volcanoes via the navigation bar.



5. The user will then choose a country to search for a volcano in. They may choose to also filter by distance, but this is optional.



6. The user may now interact with the map. They can then press the back button in the bottom left corner to view another volcano, or the logout button in the top right corner, where they will be redirected to the login screen.



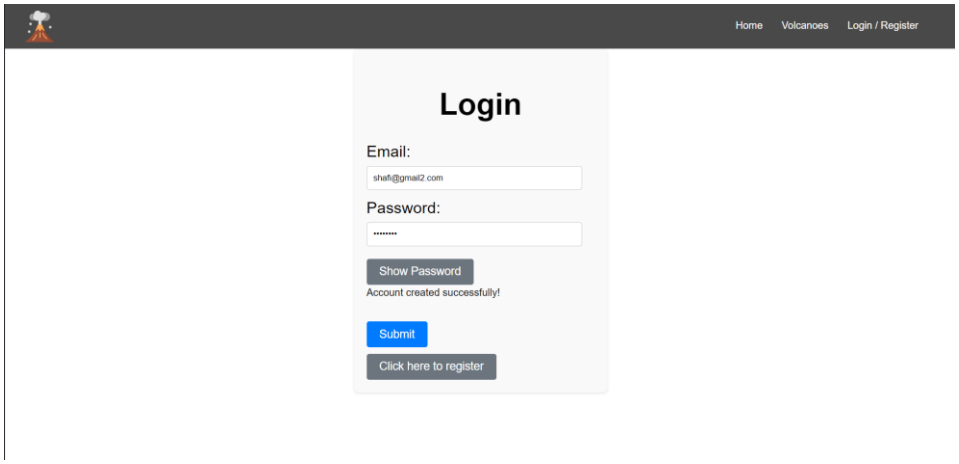
References

AudioEye. (n.d.). *Color Contrast Checker | Free Accessibility Tool | AudioEye*. Retrieved from www.audioeye.com: https://www.audioeye.com/color-contrast-checker/?cc_gate=check&utm_adgroup=CCC-WCAG&utm_source=google&utm_medium=paidsearch&utm_campaign=Generic-Color-Contrast-Checker-97F&utm_term=wcag%20contrast%20checker&utm_content=colorcontrastchecker&hsa_acc=3628656

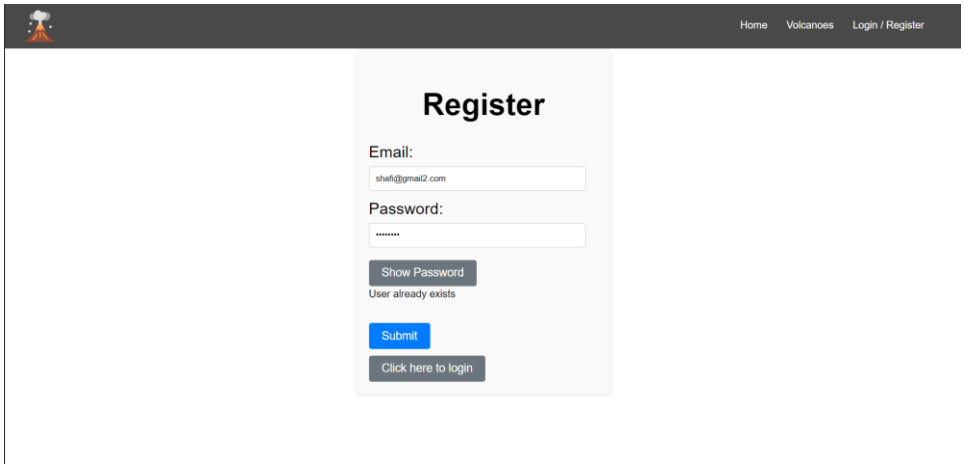
Appendix

Item 1

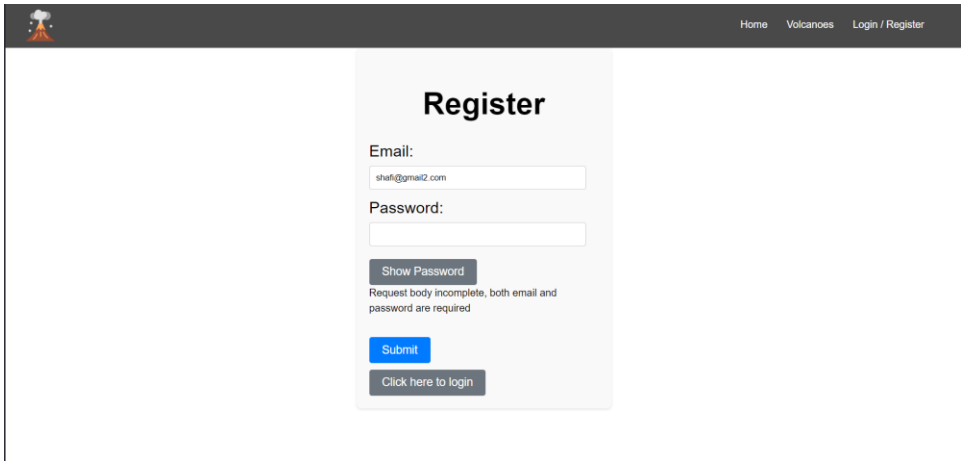
Note: user is immediately directed to login section of website upon successful registration. The message shown below show password is displayed to acknowledge successful registration.



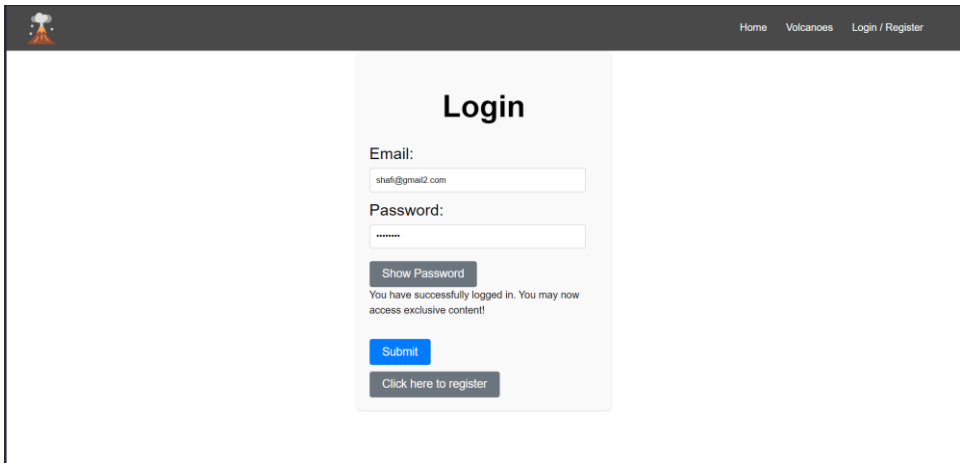
Item 2



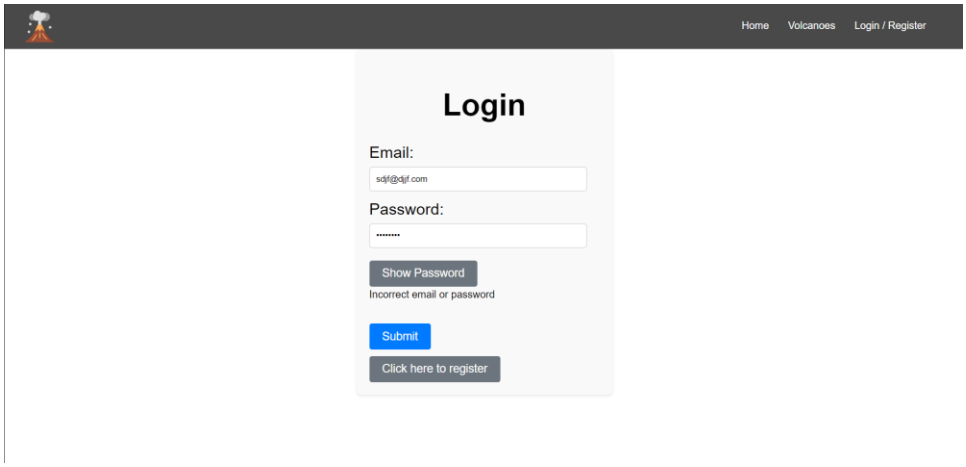
Item 3



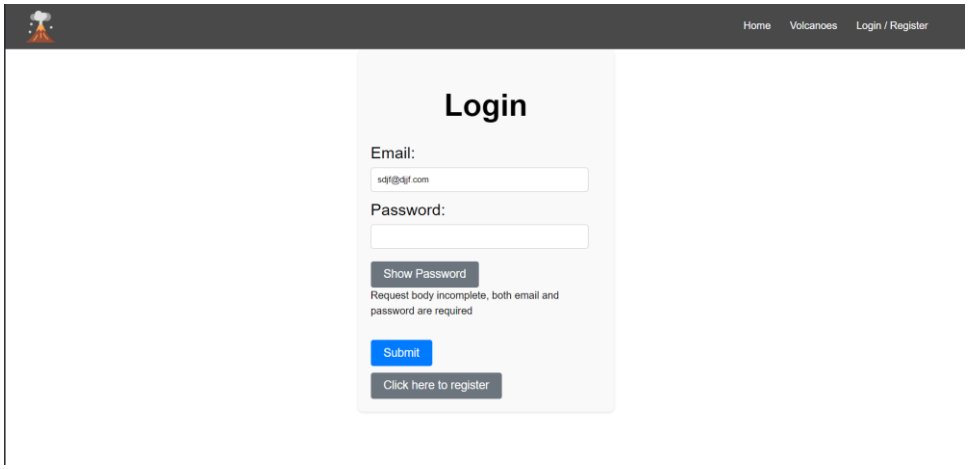
Item 4



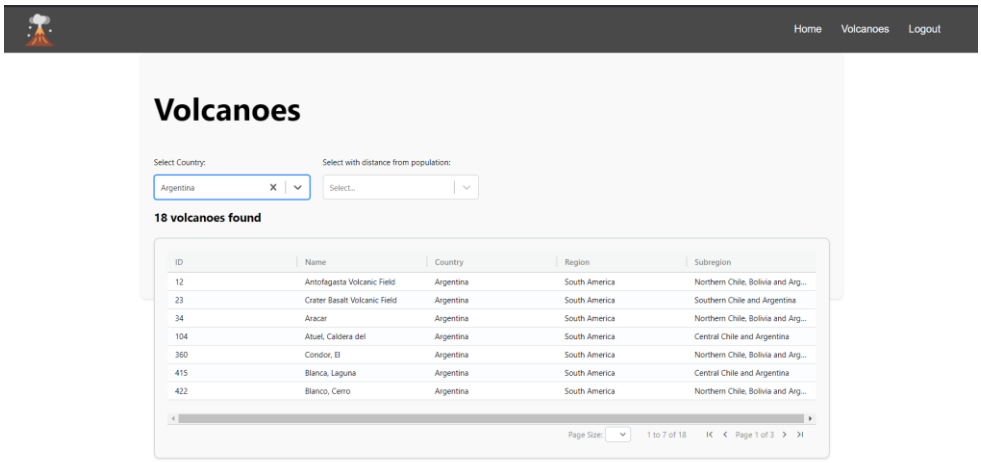
Item 5



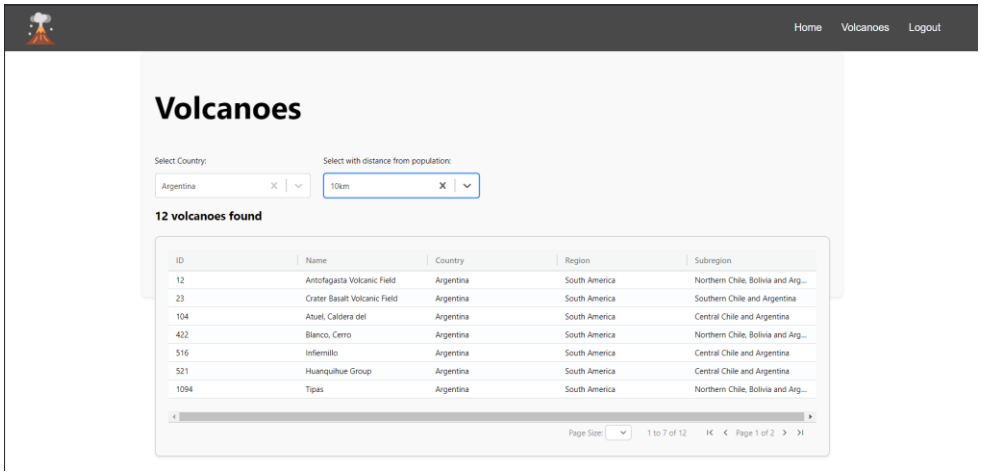
Item 6



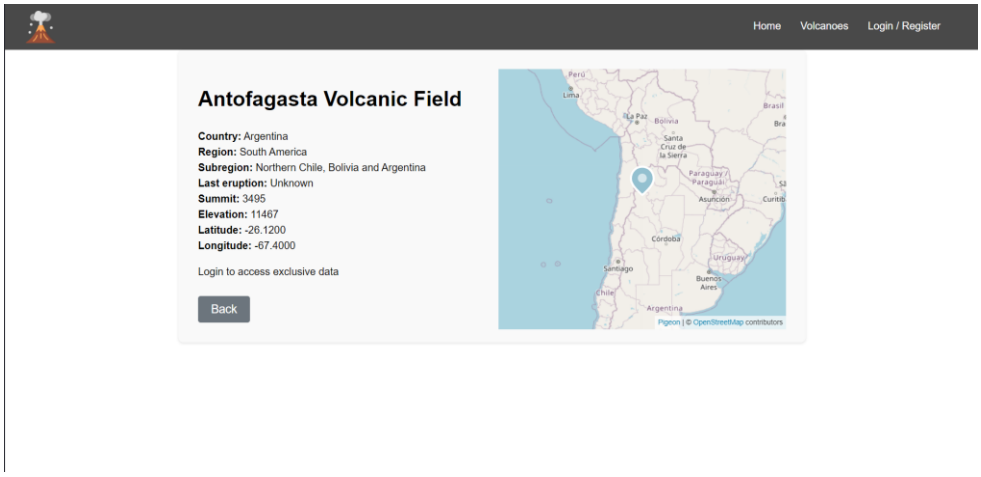
Item 7



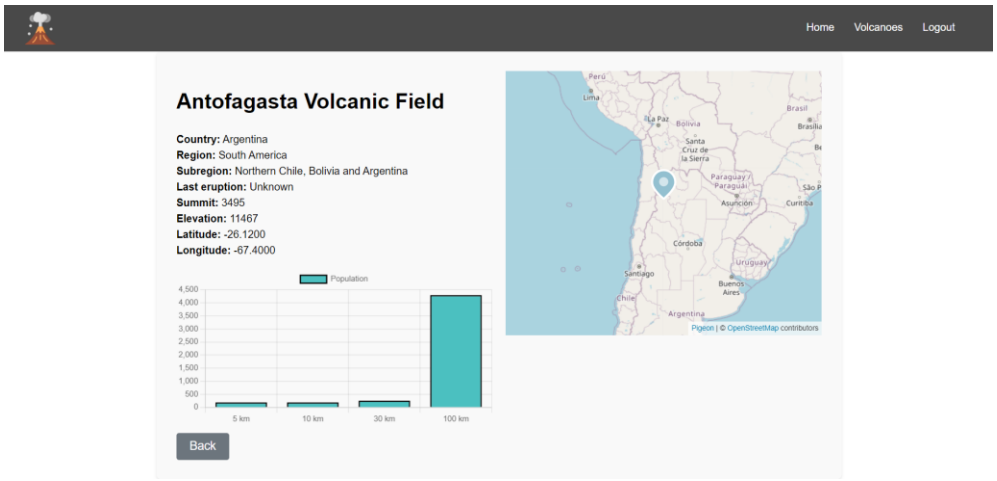
Item 8



Item 9

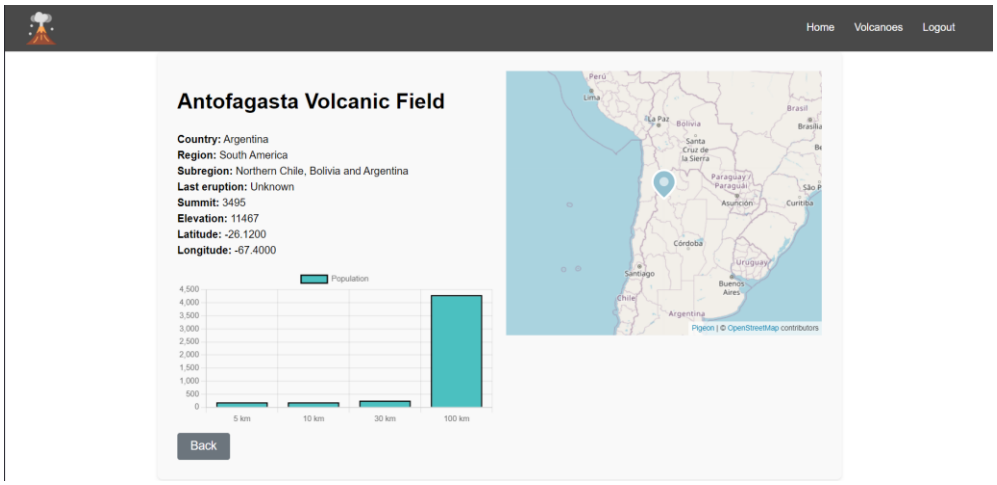


Item 10



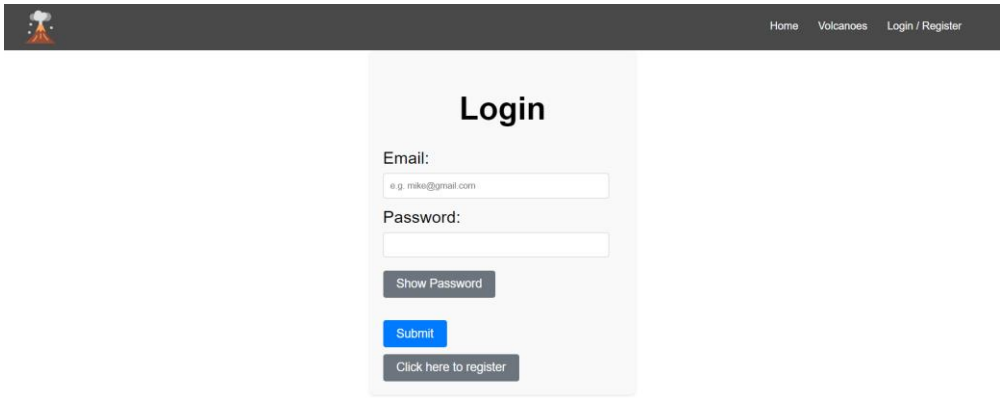
Item 11a

Before pressing logout button:



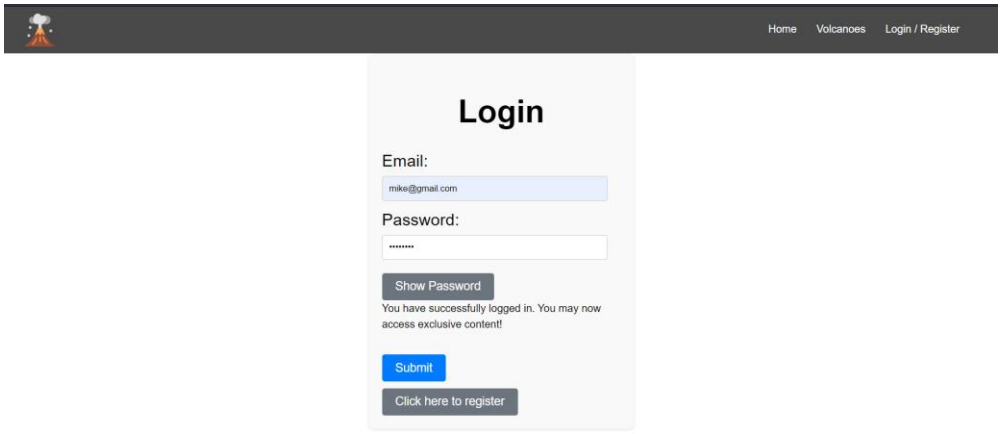
Item 11b

After pressing logout button



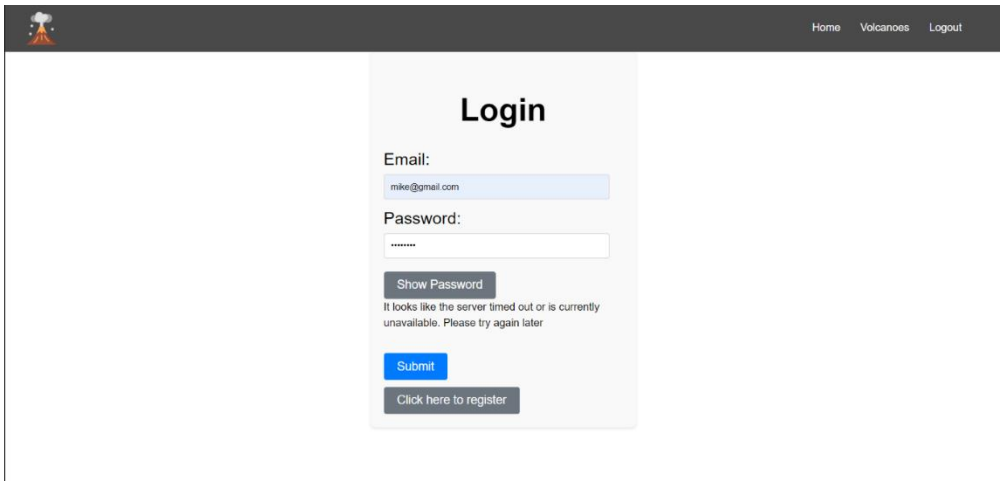
Item 11c

Logout button does not appear as soon as user is logged in. Hence test 11 is considered as a partial fail

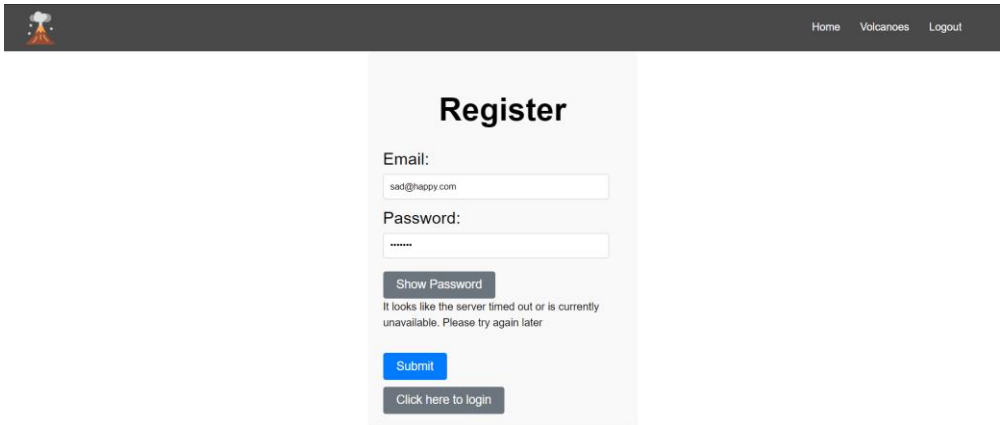


Item 12

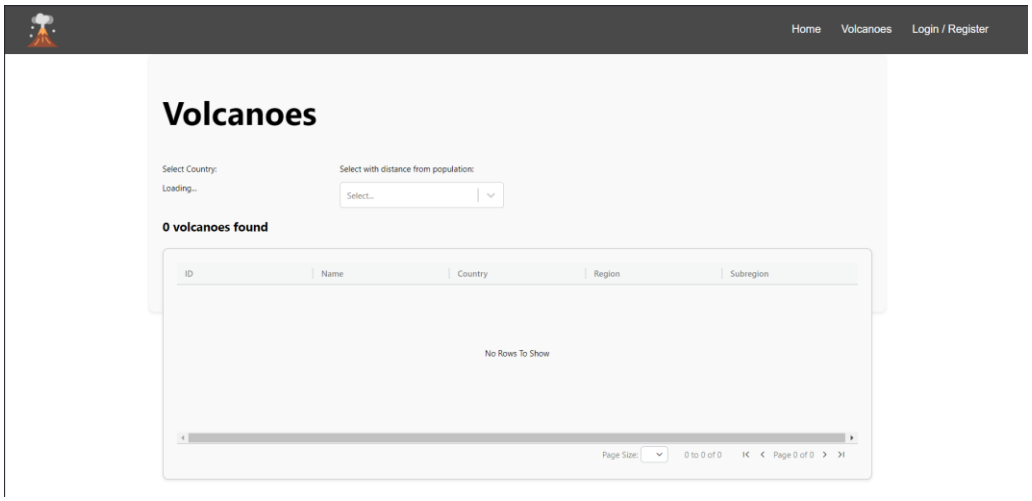
Application waited about 10 seconds until fetch attempt timed out



Item 13



Item 14



Item 15

Please sign in again. Your token has probably expired, or there is a server side error.