

**CST3130 Advanced Web Development with Big Data**

**Coursework 1: Price Comparison Website**

**Final Submission**

**Dereck Lam Hon Wah**

**M00826933**

**Date of Submission: 14.01.22**

**Lab Tutor: Waseemah Moedeen**

Table of Contents

[Table of Figures 2](#_Toc156138345)

[Introduction 3](#_Toc156138346)

[Features 3](#_Toc156138347)

[Navigation Bar 3](#_Toc156138348)

[Homepage 3](#_Toc156138349)

[Result Page 4](#_Toc156138350)

[Product Page 5](#_Toc156138351)

[Technology Stack & Explanation 6](#_Toc156138352)

[List of Websites and URLs to be scrapped. 7](#_Toc156138353)

[Database Design 8](#_Toc156138354)

[RESTFUL web service. 8](#_Toc156138355)

# Table of Figures

[Figure 1 Navigation Bar 3](#_Toc156138356)

[Figure 2 Home Page 3](#_Toc156138357)

[Figure 3 Result Page 4](#_Toc156138358)

[Figure 4 Result Page with Pagination Feature 5](#_Toc156138359)

[Figure 5 Product Page Without Comparison 6](#_Toc156138360)

[Figure 6 Product Page with Comparison 6](#_Toc156138361)

[Figure 7 Final Database design 8](#_Toc156138362)

[Figure 8 GET request 1 9](#_Toc156138363)

[Figure 9 GET request 2 9](#_Toc156138364)

[Figure 10 GET request 3 9](#_Toc156138365)

# Introduction

"MoLarak" is a minimalist website for comparing prices of alcoholic beverages, aimed at assisting users in effortlessly finding top deals on a range of alcoholic drinks across multiple websites. It offers an intuitive interface where users can search for their preferred alcoholic beverages, explore a selection of recommended featured products, and obtain comprehensive product details along with links to numerous websites for completing their purchase.

Features

### Navigation Bar

The navigation bar present on all the different web pages and consists of “MoLarak” logo to redirect the user back to the homepage.

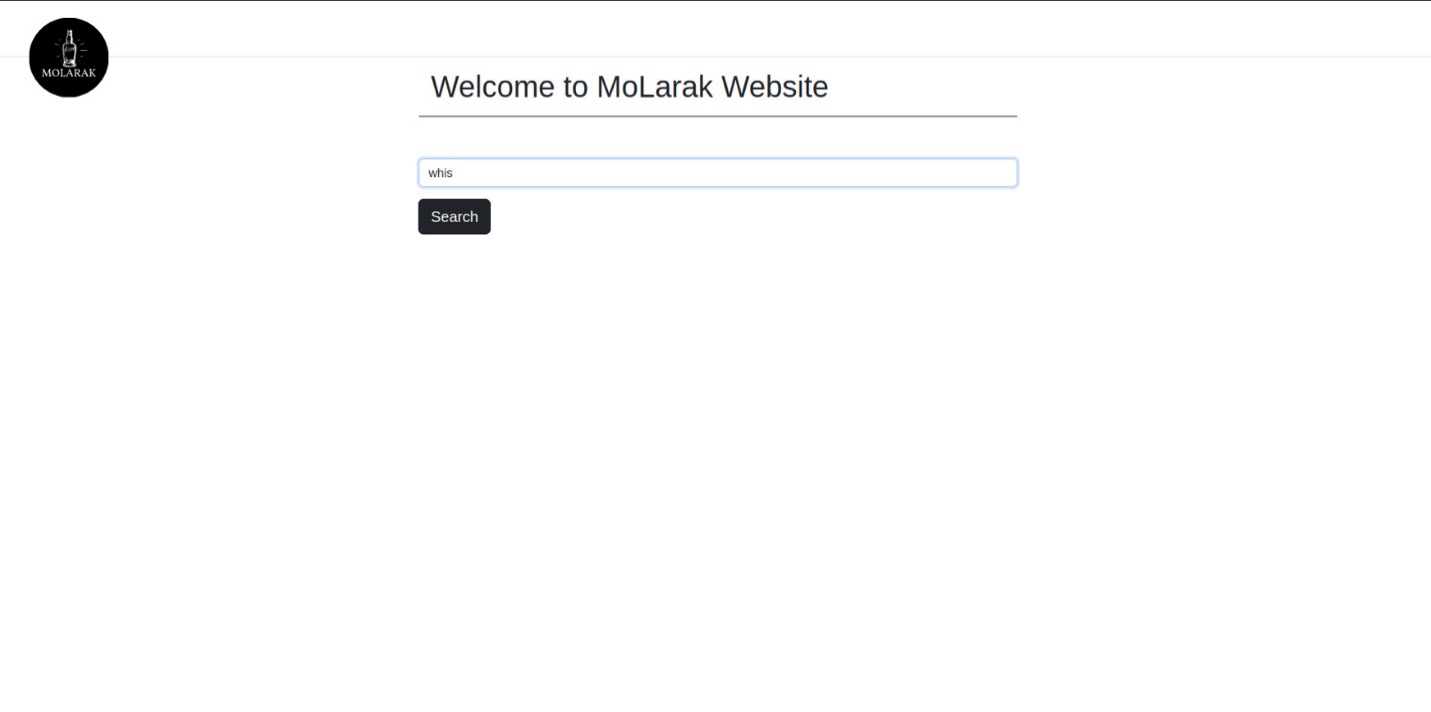


Figure 1 Navigation Bar

### Homepage

On the home page, there is a search bar to enter the name of the desired alcoholic beverage. To enhance user experience and prevent empty queries, the search functionality has been designed with specific constraints. If the search bar remains empty, a search button will not be displayed and additionally, pressing the 'Enter' key will not trigger a search action, ensuring that the user won’t proceed to the results page without a query entered.

Upon entering a valid query, the user will automatically be directed to the result page through the following link: /alcoholic-drinks/search/{{searchQuery}}?limit=9&offset=0.

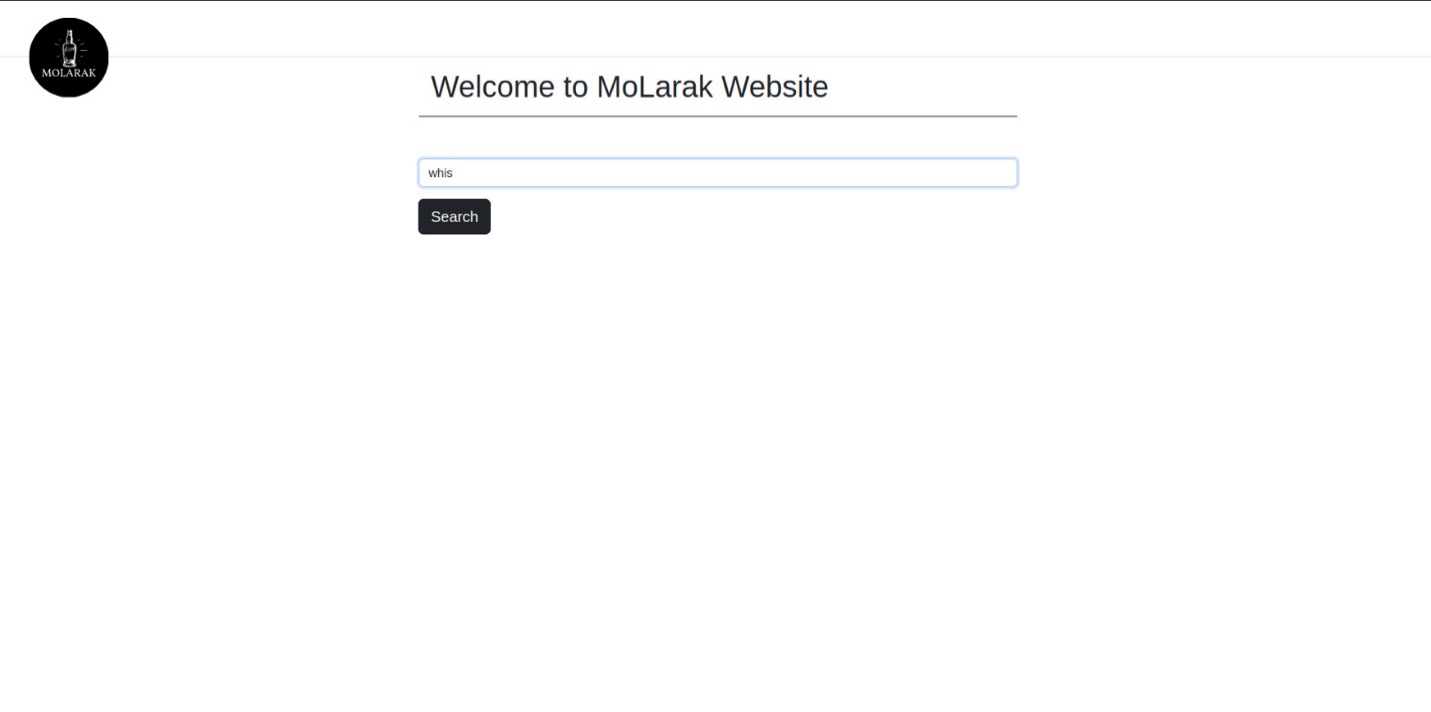


Figure 2 Home Page

### Result Page

Upon page creation, a GET request is sent to the MySQL server with the following endpoint: /alcoholic\_drinks/search/:search?limit=9&offset=0. This request is used to perform a partial search to retrieve a list of alcoholic drinks, limited by the offset and limit parameters and a complete count of the partial search results.

The retrieved list of alcoholic drinks is dynamically displayed in a card-style format, which includes the alcoholic drink image, brand, name, and a "compare" button to redirects the user to the Product page for more details and to display the product comparison results.

To enhance user experience and prevent latency during information retrieval and DOM rendering, the list of results is stored in a container with an appropriate pagination functionality at the bottom. To interact with this functionality the user can click on five respective buttons: "Previous," "Next," "Current page," and two buttons representing a page before and a page after the current page. Each of these actions triggers the same GET request, but with a modified offset value, allowing the user to access a different list of products.

Thanks to the Vue computed and watcher functions, we have ensured a seamless experience during the pagination process, by preventing the entire webpage from reloading; instead, only the container holding our results is refreshed.

If the search query doesn't return any alcoholic drinks, the webpage will display an appropriate message to inform the user of the lack of results. Upon clicking the “Compare” button, the user is automatically directed to the product page through the following link: /alcoholic-drinks/{{id}}.

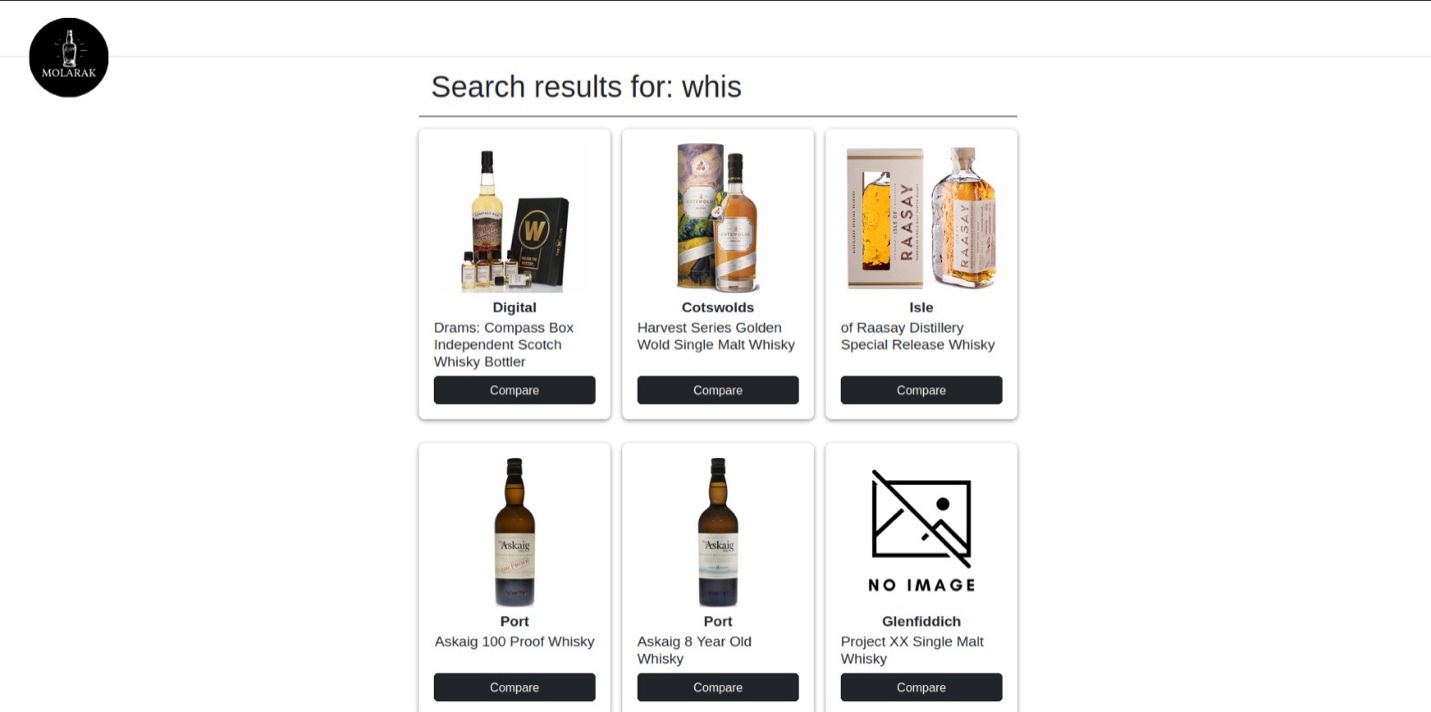


Figure 3 Result Page

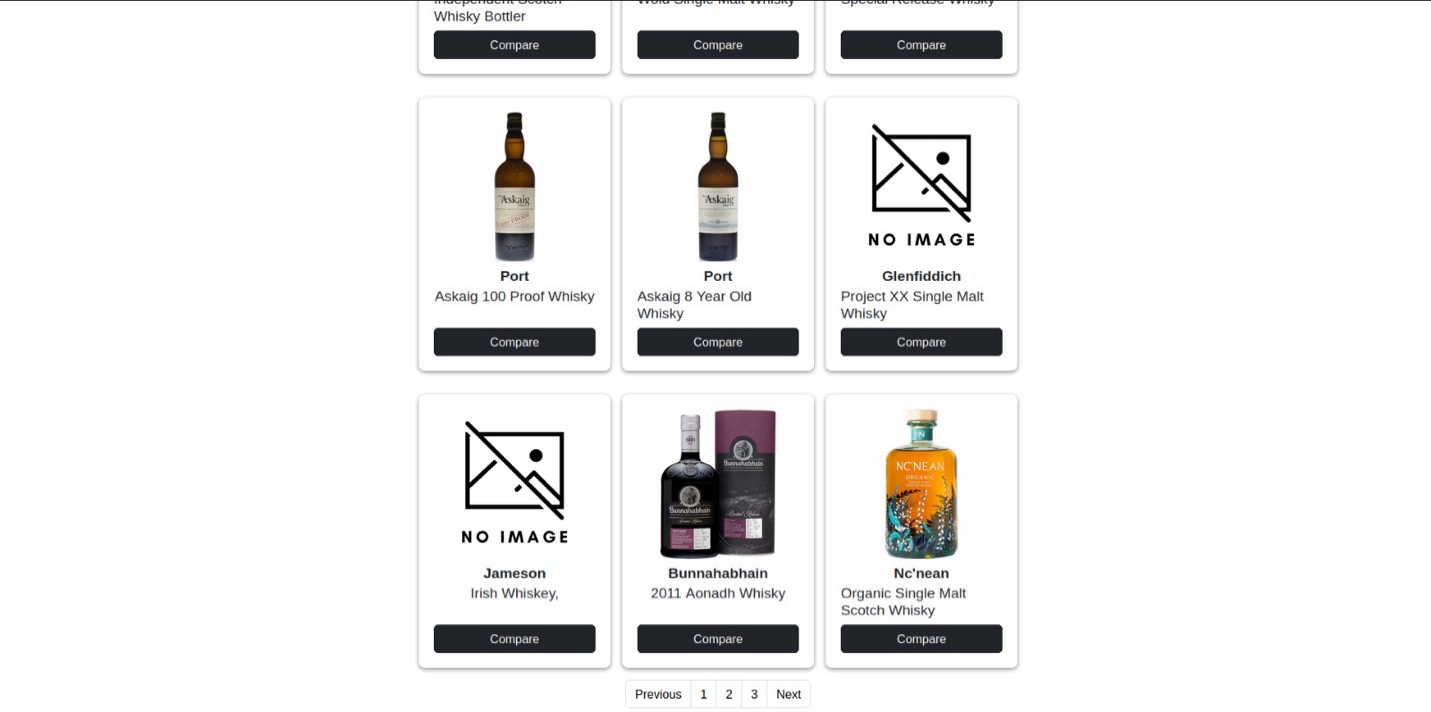


Figure 4 Result Page with Pagination Feature

### Product Page

Upon page creation, two GET requests are initiated towards the MySQL server. The first request, alcoholic\_drinks/:id, is used to fetch comprehensive information specific to the selected alcoholic drink. The second request, alcoholic\_drinks/comparison/:id, is employed to retrieve details about other alcoholic drinks that share the same name as the selected one.

The product page comprises two distinct components. The first component is responsible for presenting all the relevant details associated with the selected alcoholic drink. It also features a button that allows users to navigate to the source web URL of that product.

The second component is dynamically rendered, appearing only when additional alcoholic drinks are found in the database. This component serves as a list of alcoholic drinks displayed in a card-style format, enabling users to compare prices, and a quick redirection to the respective product source pages.

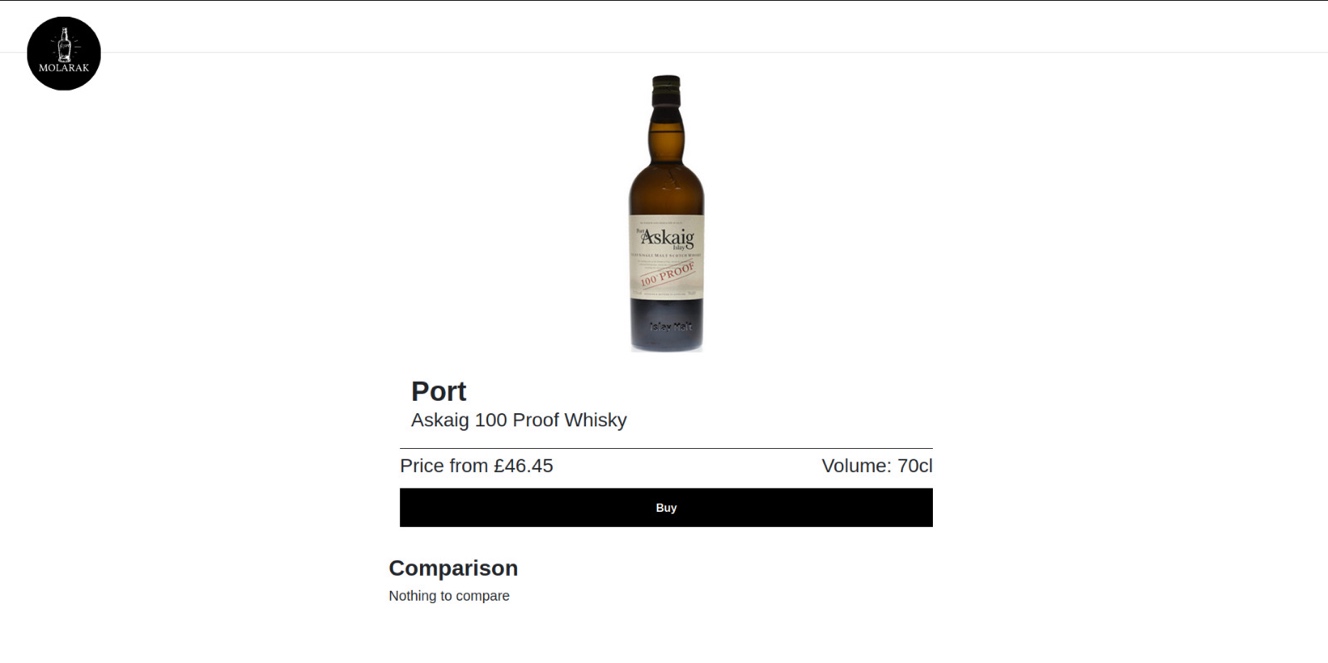


Figure 5 Product Page Without Comparison

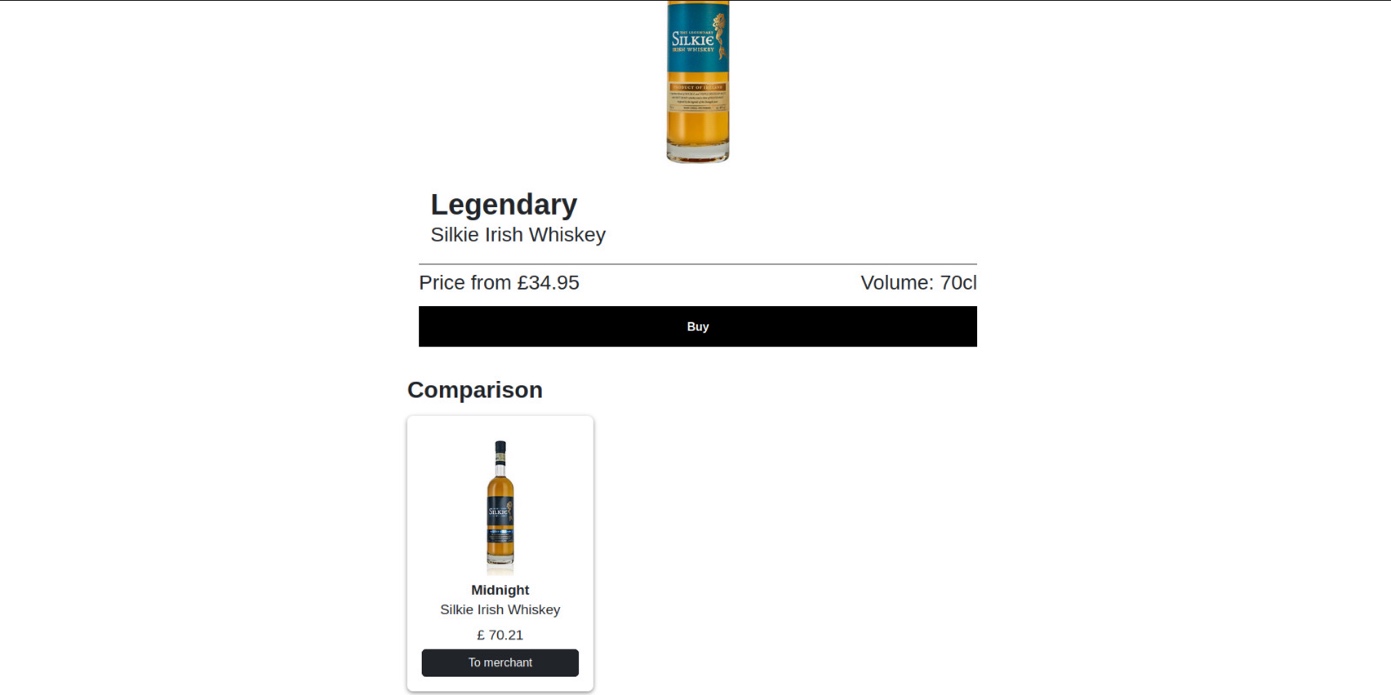


Figure 6 Product Page with Comparison

## Technology Stack & Explanation

The website was created by utilizing both front-end and back-end technologies in order to provide users with a smooth browsing experience:

|  |  |  |
| --- | --- | --- |
|  | **Technologies** | **Description** |
| **Front-end** | HTML, CSS, JavaScript and Vue3 | To create an engaging and responsive user interface through the application of a design pattern MVVM (Model-View-View-Model) to separate data and views. The View layer will listen to the property changes and renders them in the virtual DOM. |
| **Back-end** | JavaScript using Node.js and Express.js | To handle server-side operations efficiently |
| **Database** | MySQL or MariaDB | A SQL database to store data related to alcoholic drinks and its comparison. |
| **API Integration** | REST API via Axios | Implementation of a newer promise-based library compared to Ajax and jQuery to asynchronously send and receive data to the backend, ensuring real-time accuracy and availability of information. |
| **Data Collection** | Java, Selenium | Using Selenium and multi-threading to asynchronously pull and download data from various alcoholic beverage provider websites. Implemented a Thread Manager to efficiently manage the five website scraper to scrape and store the information in the MySQL database. |
| **Build and Dependency Management** | Maven | To build Java software, manage project dependencies, and automate the build process using the pom.xml file. Also using Java doc comments, to generate the Java Project documentation. Implementation of Log4J framework for a structured logging interface. |
| **Data Persistence** | Hibernate | To handle object-relational mapping using annotation and simplify database operations for storing and managing data in the SQL database. |
| **Dependency Management** | Spring Framework | To manage the dependencies between various DAO Java classes, enhancing modularity and making the application more maintainable. |
| **DAO Direct Access Object Layer** | Hibernate and Spring Framework | Bootstrapping Hibernate 5 SessionFactory with the native Spring Hibernate API to implement a DAO layer for separation of the business logic and the database operations, making the scraper application more modular. |

# 

## List of Websites and URLs to be scrapped.

1. The Whisky Shop - [www.whiskyshop.com](http://www.whiskyshop.com)
2. Master of Malt - [www.masterofmalt.com](http://www.masterofmalt.com)
3. Onbuy – [www.onbuy.com](http://www.onbuy.com)
4. The Whisky Exchange - [www.thewhiskyexchange.com](http://www.thewhiskyexchange.com)
5. Spirits Kiosk – spiritskiosk.com

## Database Design

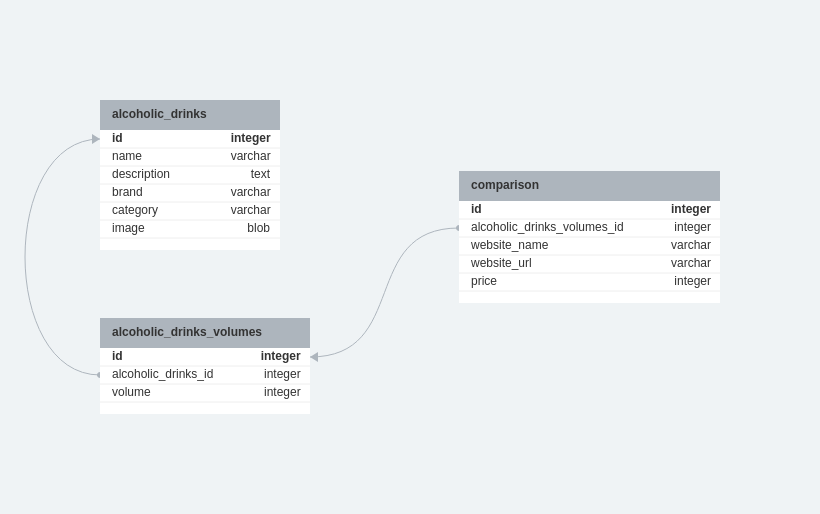


Figure 7 Final Database design

## RESTFUL web service.

A RESTful web service is an architectural approach for designing networked applications, where resources are represented as unique URLs (Uniform Resource Locators) and can be manipulated using a standard set of HTTP methods. In our application, we implemented a JavaScript backend using node.js and express.js, and used AXIOS for communication between the frontend and RESTFUL web service. Below find the endpoints and their results:

**GET /alcoholic\_drinks/search/:search?limit=9&offset=0** – perform a partial search to retrieve a list of alcoholic drinks, limited by the offset and limit parameters and a complete count of the partial search results.

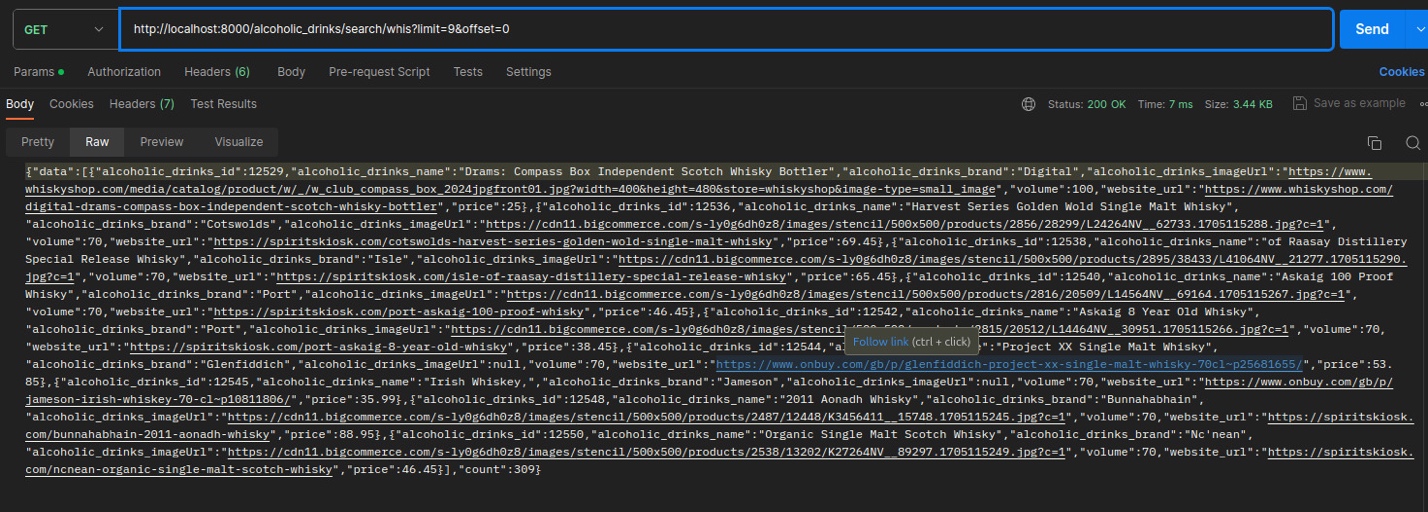


Figure 8 GET request 1

**GET alcoholic\_drinks/:id** - to fetch information specific to the selected alcoholic drink.

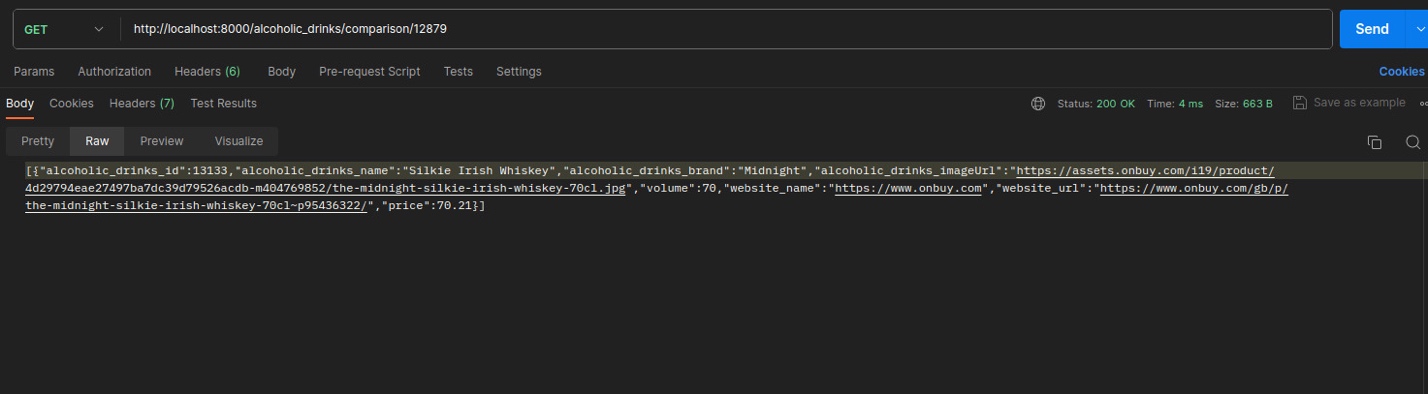


Figure 9 GET request 2

**GET alcoholic\_drinks/comparison/:id** - to retrieve details about other alcoholic drinks that share the same name as the selected one.

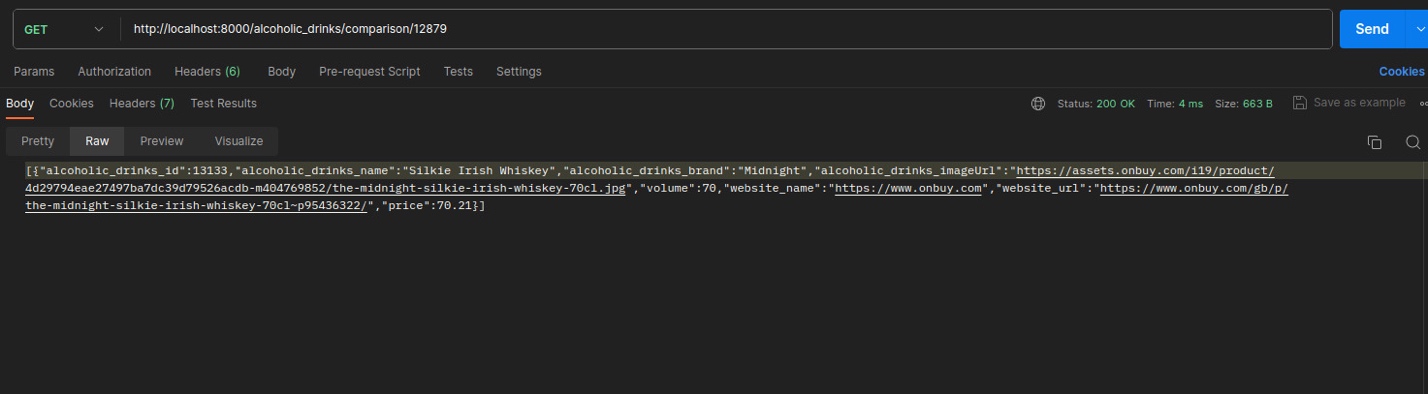


Figure 10 GET request 3