Version 1.

**Jul 8, 2022**

CAR SELLING WEBSITE

SYSTEM

REQUIREMENT

**SPECIFICATION**

**Author:** Dao Minh Hieu

**Security Classification:**

110 Bishopsgate, London EC2N 4AY

Tel: +44 (0)20 7333 0033

Email: [info@nashtechglobal.com](mailto:info@nashtechglobal.com)



Table of contents

[1. Overview 2](#_Toc108165968)

[**1.1** Purpose of this document 2](#_Toc108165969)

[2. Functional requirements 2](#_Toc108165970)

[**2.1** Search cars 2](#_Toc108165971)

[2.1.1 Description 2](#_Toc108165972)

[2.1.2 Techniques 2](#_Toc108165973)

[**2.2** Register sell car 2](#_Toc108165974)

[2.2.1 Description 2](#_Toc108165975)

[2.2.2 Techniques 2](#_Toc108165976)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Document Version History | | | | | |
| **Version** | **Effective Date** | **Author** | **Details** | **Reviewer** | **Approvers** |
| 1.0 | Jul 08 2021 | Dao Minh Hieu | Created | All members | Le Cao |

Confidentiality

This document is distributed on a restricted basis, is commercial in confidence to the recipient, and may not be used for any purpose other than that associated with a NashTech project. The contents of this document may not be disclosed to any third parties without the expressed advance written authorisation of NashTech.

# Overview

## Purpose of this document

This document provides the functional requirements covered in the project “Car Selling Website”. All the features will include a feature name, a short description and React and Next.js techniques that the developer are applying to their works.

# Functional requirements

Describe some main functional requirements and how the solution has covered each of the items.

## Get All Products

### Description

* As a client / user when they go to the home page then they can see all products existing in the system.

### Techniques

* To take more use SSR for SEO optimization, all items will be loaded at build time via the getStaticProps function. Following that, the product list will be supplied as a prop to the Product component and saved as a state of that component.
* That state will be used by the component to render all of them on the browser.

## Search cars

### Description

* On the hone page, users can search for cars by typing keywords into the search box.
* The system allows searching by name, brand, model, and year.

### Techniques

* Create two components named SearchBox and CarList
* In the index page, use “Static Site Generation” to pre-render the list of cars at build time, which is good for SEO.
* The index page contains the SearchBox and the CarList components. Whenever the user triggers a search, the index page calls the API to get the cars then passes to the CarList to render.
* Use the State hook to store the current cars.

## Register sell car

### Description

* Users want to sell their cars online and they need a place to post information to sell them.
* To advertise cars for sale, user must be signed in and go to sell car page by pressing sell car button. On the sell car page, the user must fill out all of the information required to sell the car and submit the form, after that the selling car will appear on the homepage where the buying car can find them.

### Techniques

* Use functional components to create question field components (TextField, AutoComplete, DatePicker…) each field are required value, if the input is invalid, the error will be displayed. These components are built with “@mui/material” library.
* The form data after submission will send to API by using fetch API.
* Use useSession() to check user is signed in after reaching the register sell car page.

## Filtering car by brand

### Description

* In addition to the standard filter, users are able to narrow the scope of search result to a specific brand.
* The brand filter will be presented in form of a carousel with list of brands per page. The carousel is allowed to swipe left or right to reveal other pages. Whenever users pick up a brand, they will be navigating to a sub-page for that brand with list of belonging cars.

### Techniques

* Taking advantage of SSR for SEO optimization, the brand selector and sub-page for each brand will be generated at build time.
* For brand selector component, list of brands will be fetched at build time to render all pages of carousel. A third-party library [Swiper](https://github.com/nolimits4web/swiper) for Reactjs is used to render and bootstrap carousel at client-side.
* For sub-page per brand, dynamic routes is applied to handle all brands, or pages in equivalent, then prerendered all the pages at build time. Brands will also be retrieved at build time to identify all possible value. The default search result, which is list of cars belongs to a specific brand will be processed at client-side, using common Reactjs features such as Component, Hooks(useState, useEffect).