

# Car Rental System – Specification

## Introduction

In this project, you are required to develop a dynamic website with interactive features for a car rental company. The website should provide a streamlined online renting experience for potential users. It should be able to handle various exceptions, such as input validation and change in car availability.

The website should enable users (i.e., potential customers) to

- Browse cars by type or brand,
- Search for cars using keywords,
- Check a car's details and availability (whether it is rentable),
- Place a rental order (reservation) for an available car,
- Confirm or drop a rental order.

This assignment accounts for **35%** of the total mark in the subject. You are required to complete this assignment individually.

## Project Objectives

Creating a car rental application using a combination of scripting languages, descriptive language, and developing tools, including

- Designing professional looking websites with rich interactive features,
- Devising customised data storage/persistence using JSON, XML, SQL or NoSQL,
- Using AJAX techniques to create dynamic and interactive web applications without full page reloads.

## Functional and Visual Requirements

### 1) Website logo

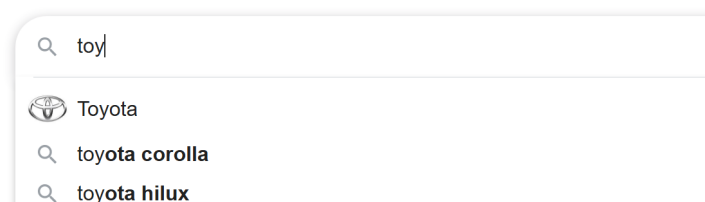
There should be a logo for the website. The logo should appear on all major pages of the website.

User can click the website logo at any time to get to the website's homepage.

### 2) Search box

There should be a search box allowing users to look up cars by using keywords related to type, brand, car model and description.

The search box should show **real-time suggestions** upon user input – something like below.



A screenshot of a search box with a magnifying glass icon on the left. The input field contains the text "toy". Below the input field, there are three suggestions, each preceded by a magnifying glass icon: "Toyota", "toyota corolla", and "toyota hilux".

### 3) Search filters

Car type (e.g., Sedan, Wagon, SUV) and brand (e.g., Ford, Mazda, BMW) can be used to filter search results.

The filters can be used with the search box: filters can be applied to search results; search can be conducted within the cars specified by the filters.



A screenshot of a search interface. It features a text input field with the placeholder text "What are you looking for?". To the right of the input field is a dropdown menu labeled "CATEGORY" with a downward arrow. Further right is a blue button labeled "SEARCH".

### 4) A grid view of cars

When multiple cars are shown on one page, they should show in a grid layout.

Every car should display some key (not necessarily all) information about [car type](#), [brand](#), [model](#), [image](#), [year of manufacture](#), [mileage](#), [fuel type](#), [rental price per day](#), [availability](#), and an optional [description](#).

Every car is uniquely identified by its Vehicle Identification Number (VIN), a 17-character alphanumeric code that acts as the vehicle's unique fingerprint. The VIN is invisible to users on the website.

For each available car, user can click a “rent” button (or other clickable item) next to the car to be redirected to the reservation page.

**A user can only reserve one car (identified by VIN) at a time** – If the user clicks multiple cars, only the last car's information will be loaded to the reservation page.

## 5) Reservation

Upon entering this page, user sees all information about the car that he/she last clicked.

User can place a rental order if the car is still available. To place an order, user must provide his/her [name](#), [phone number](#), [email](#), and [driver's license number](#). The user must also specify the [start date \(i.e., first day of rental\)](#) and [rental period \(by number of days\)](#), so the page can calculate and show an estimated [total price](#) for the order. At this point, user can proceed to submit the order (e.g., by clicking a “submit” button).

If user changes his/her mind and decides not to proceed with the order, the user can either **click the “cancel” button** or **just leave the page** (e.g., by closing the page or heading to another page). In the first case, the website will clear user's input and take him/her to the homepage. In the second case, the website will save user's input in the reservation form (if there is any); so, the next time the (same) user see the form, the website will pre-fill the reservation form with the saved input from the last time.

## 6) Order confirmation

Once an order is placed, the page shows a clickable item (e.g., a web link), which user can click to confirm the order. The order will only make the car unavailable but will not go through until the user clicks the link.

This confirmation will trigger an update on the order status from “pending” to “confirmed”.

## Data Persistence Requirements

- There should be a minimum of two data structures to store information about **cars** and **rental orders**.
- Use JSON, XML, MySQL or comparable techniques (e.g., NoSQL) for data persistence — we have provided two JSON files to exemplify how the car and order information can be stored as files instead of in database tables.
- We recommend using Cookie, Session, localStorage or sessionStorage to temporarily store user input in the reservation form when user leaves the reservation page without cancelling the form.

## Front-end Techniques Requirements

- You are required to use AJAX, jQuery or other comparable techniques (with AJAX features) to enable asynchronous data retrieval and dynamic updates.

## Deployment Requirements

- You are required to deploy your website to Elastic Beanstalk or alternative cloud platforms, allowing remote access to your website by tutors.

## Website Presentation (or Overall Looking) Requirements

- All texts and labels are easy to read.
- Text fonts used on the web pages are contemporary and of appropriate font sizes.
- Everything on the pages has the appropriate font colours and background colours.