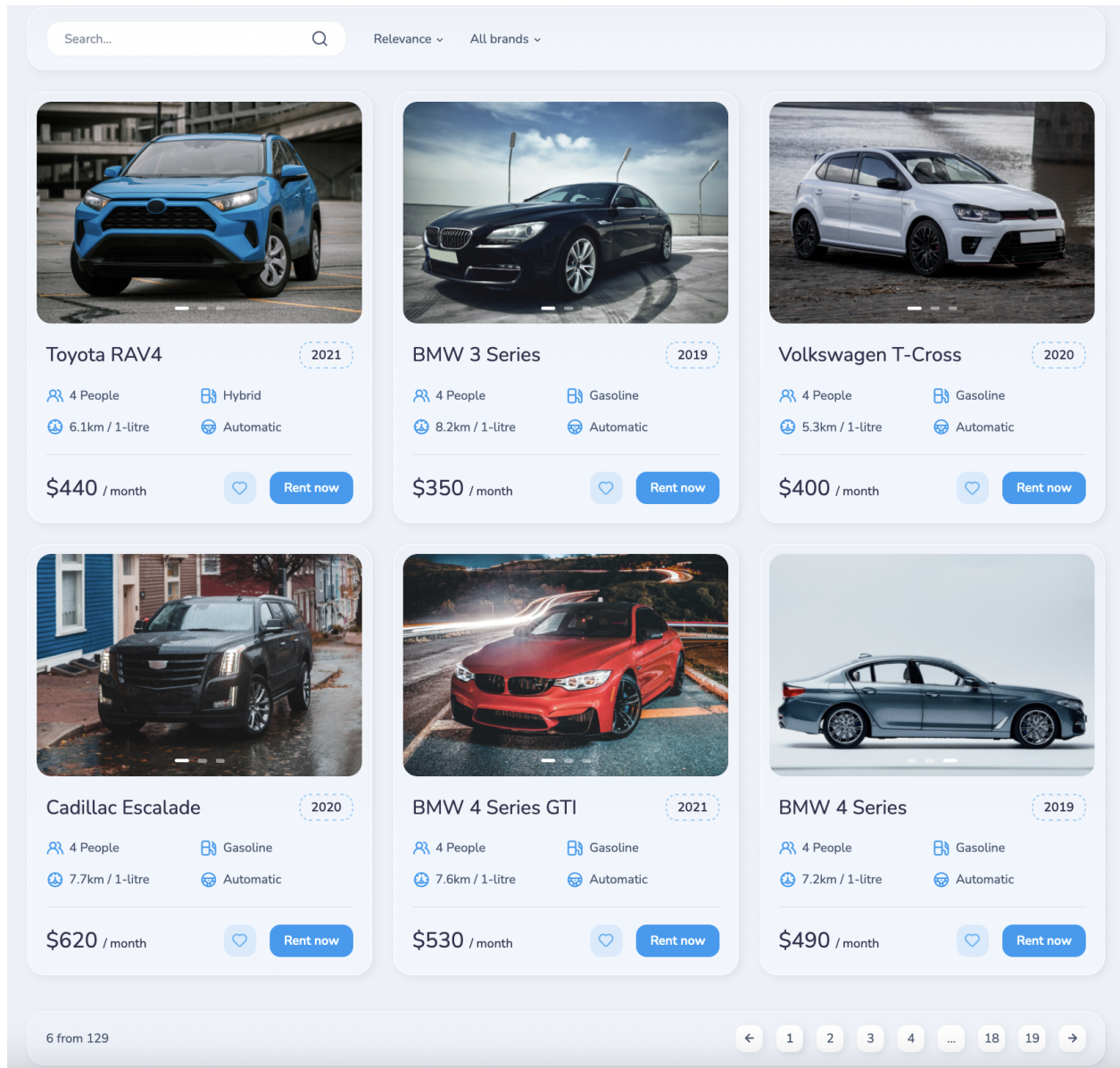# Seller App

Frontend Take-Home Assignment

Task

Create a car search website in Reactjs for desktop-only screens, you don't have to make it responsive.

- Cars can be searched and paginated through. You can ignore the Relevance and All Brands filter.
- Car cards are listed 3 on each row and a time 6 cards should be shown
- Each car card should be designed as it is seen in the image (not pixel-perfect but should look similar)
- The search should work on the car name.
- On the pagination, the URL should change to *page/:page* for example. page/1 -> default, page/2, etc.
- Show only 10 pages for the pagination, it should be hard coded 10, and should not depend on the car count.
- Pagination should have the next and previous buttons along with the numbers and it should be fully functional

Search... 　Relevance ⌄　All brands ⌄

Toyota RAV4　2021
4 People　Hybrid
6.1km / 1-litre　Automatic
$440 / month　Rent now

BMW 3 Series　2019
4 People　Gasoline
8.2km / 1-litre　Automatic
$350 / month　Rent now

Volkswagen T-Cross　2020
4 People　Gasoline
5.3km / 1-litre　Automatic
$400 / month　Rent now

Cadillac Escalade　2020
4 People　Gasoline
7.7km / 1-litre　Automatic
$620 / month　Rent now

BMW 4 Series GTI　2021
4 People　Gasoline
7.6km / 1-litre　Automatic
$530 / month　Rent now

BMW 4 Series　2019
4 People　Gasoline
7.2km / 1-litre　Automatic
$490 / month　Rent now

6 from 129　← 1 2 3 4 ... 18 19 →

Guidelines

- You can use any framework of choice but Reactjs is preferred
- You can use CSS frameworks like Bootstrap and Tailwind
- The code should be functional and each listed action should happen in order to pass
- It is good to use a global state management library like Redux

- Follow best practices.
- Commit the code to a repository and include a readme. Share the link to the repo once done.
- Use mock data from a JSON file, API integration is not required.
- Make sure the routing is there and the pagination is working properly

Evaluation

- We will evaluate how you structure and store the data in JSON
- How do you create the components and re-use and compose them
- How data is shared between the components.
- How well does the pagination work?

Add on

- If unit test cases can be added.