Lastminute.com Exercise

Parking Widget

Dev environment: feel free to use any (Bower, CommonJS, Grunt, Gulp, NPM, Webpack,...)
CSS: SASS/LESS permitted, no other frameworks allowed
JS: VanillaJS is mandatory (no frameworks/boilerplate such as Angular, React,… should be used) but you can use Underscore/Lodash and jQuery, any template engine (Handlebars, Mustache,…) and testing tools (Karma, Jasmine, Mocha,...).

Scenario

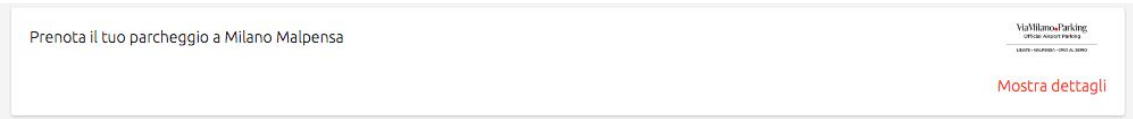
You are going to develop a parking widget:

- "Parking" because the goal is to show a collection of parking slots and let the user buy one
- "Widget" because it should be an independent module that can be embedded in any Web application

You are asked to implement the widget UI composed by a vendor area and a parking slots area.

1. The vendor area should list the features of the vendor in a togglable manner.

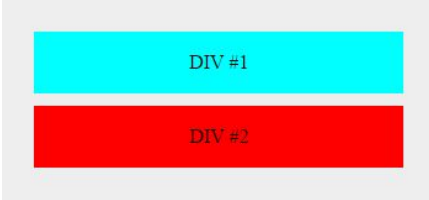
Closed example:



Opened example:

2. The parking slot area contains all the parking slots.
   - You should design parking slots as “blocks” aligned differently between desktop and mobile devices: horizontally on desktop devices and vertically on mobile ones.
   - Inside each parking slot you can insert the info about the parking slot itself
   - A parking slot should be selectable by the user

Mobile example:



Desktop example:

Notes:

- In the source code you have a *model.json* with the data from the server. Use them to fill the UI, as mock data received from your server-side service
- Clicking one of the parking slot in the Widget will add the parking slot to the shopping cart of the host Web app
- Add a button show/hide details (the map and the features list) of the selected parking slot
- You can use the same layout proposed in the exercise, but feel free to change it if necessary

Before start implementing

Think about widget interface requirements.
The Web app that wants to use the widget must respect/implement an agreement between the Web app and the widget in order to retrieve the parking slot that the user has selected inside the widget.

1) How can you initialize and render the widget in that Web page? How can communicate the Web app with the Parking widget (and vice versa)?

2) How can you guarantee that another developer can modify your code without breaking the features? How can you guarantee to modify your code during a refactoring without breaking any functionality?

3) How do you differentiate desktop and mobile devices?
Write your thoughts.