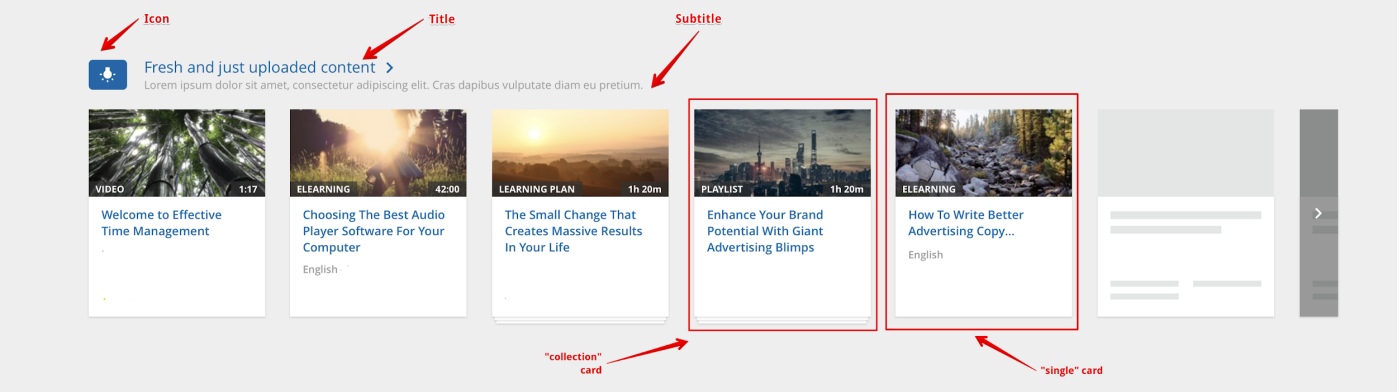
Frontend Developer assignment

Carousels are UI elements used to represent collections of items (cards, pictures, ..) in a user-friendly and intuitive way. Users can either click on arrows on both sides or swipe horizontally to browse chunks of items inside the collection.

The candidate is asked to implement a reusable carousel component, strictly following the specifications reported below.



The candidate is expected to submit a project composed by the following files at least:

- index.html, the demo page that instantiates one or more instances of the carousel component. Please, follow the structure as below:

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Carousel demo page</title>
    <link rel="stylesheet" type="text/css" href="https://fonts.googleapis.com/icon?family=Material+Icons">
    <link rel="stylesheet" type="text/css" href="//css/carousel.css">
    <link rel="stylesheet" type="text/css" href="//css/styles.css">
    <script type="text/javascript" src="//js/carousel.js" />
    <!--
        ADD HERE ANY ADDITIONAL CSS OR JS YOU MAY DEEM APPROPRIATE
    -->
</head>
<body>
    <h1>Carousel demo page</h1>
    <div id="my-carousel"></div>
    <div id="my-carousel2"></div>
    <script type="text/javascript">
        var options1 = {
            container: "my-carousel",
            title: "Fresh and just uploaded content",
            subtitle: "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua",
            fetchCards: function(chunkSize) {
                // Function that returns "chunkSize" card objects to be displayed in the carousel
                // Each card object is structured as follows:
                return {
                    image: "http://writingexercises.co.uk/images2/randomimage/cyclists.jpg", // url of a random image taken from the web
                    type: "elearning", // a string with the following allowed values (video, elearning, learning_plan, playlist)",
                    duration: 3600, // a duration in number of seconds (to be converted in human readable format, see mockup)
                    title: "Welcome to Effective Time Management",
                    cardinality: "single" // Whether this card is single or a stack of cards. Possible values = "single" or "collection"
                };
                // Simulate a random delay of an API call here, to demonstrate the usage of card placeholders
            }
        };
        var carousel1 = new Carousel(options1);

        var options2 = {
            container: "my-carousel2",
            title: "Another carouse instance title",
            subtitle: "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua",
            fetchCards: function(chunkSize) {
                // see options1 for the specifications of this function
            }
        };
        var carousel2 = new Carousel(options2);
    </script>
    <!--
        ADD HERE ANY ADDITIONAL INSTANCES OF THE CAROUSEL
    -->
</body>
</html>
```

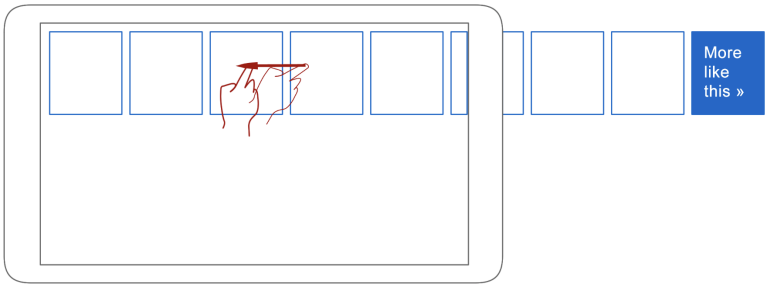
- carousel.scss, the source SCSS containing all styles needed by the carousel component.
- carousel.css, the compiled CSS for the carousel.
- styles.scss, the source SCSS with any additional style needed for the demo page.
- styles.css, the compiled CSS for the demo page.

- carousel.js, the Javascript file implementing the presentation logic for the carousel component.

Constraints

1. It must be possible to instantiate multiple instances of the carousel inside the same page. Each instance has its own info (title, subtitle, icon) and a horizontally scrollable set of cards inside it.
    - Horizontal scrolling is achieved primarily by clicking on the overlay arrows that dynamically appear on the left and right side of the carousel when mouse hover the carousel and only when there are hidden cards on that side.

    - *Nice to have (not mandatory!): enable scrolling also by swiping with the mouse and with the touch (e.g. on mobile devices) over the cards. See example below:*



    - The structure of the card object is described in the code excerpt above (inside the sample "fetchCards" function body). Please, notice the function "fetchCards" that is called by the carousel when a new chunk of items must be displayed. This function emulates the usage of a real REST API returning JSON data from a backend and should therefore introduce fake delays, before returning the next chunk of cards.

    - While waiting for the next chunk to be ready, card placeholders (grey cards as depicted in the mockup) must be visualized, to provide a visual indication to the end user.

    - The chunk size (e.g. 6 cards) is static and hardcoded inside the carousel component library.

    - The number of chunks returned by the "fake REST API" must be randomly decided inside the "fetchCards" function.

2. The candidate cannot use any framework and third party library (no jquery, prototype, scriptaculous...), with the exception of Material Icons fonts or other iconic fonts.

3. **Javascript code must be properly structured and well commented**.

4. The usage of ECMAScript 2015 new constructs (templates, classes, Promises), code building tools and unit testing is mandatory if the candidate is applying for a senior role.