House Rental

Front-End Program

Armin
Mobina

Advisor:
Aien
Hossein

Sep 26, 2021

Contents

1 Introduction

Airbnb is a website which makes money by charging guests and hosts for short-term rental stays in private homes or apartments booked through the Airbnb website.
This project is a remake of Airbnb. Even though the original website was created with React, we tried to implement different features using only Vanilla JavaScript and jQuery.

2 Product Documentation

2.1 Roles and Responsibilities

This project was a team-work collaboration assigned to Armin and Mobina.
Thus, the whole process of Defining tasks and responsibilities, project management, developing the project and documenting the progress was split between the both teammates.
'Quire' was used as a project management to keep track of the process ahead. also 'GitHub' was used to keep track of branches and commit histories and to prevent any conflicts throughout the project.

2.2 Team Goals and Objectives

The purpose of this project, was to build a responsive, accessible website that is fully functional using JavaScript.

- Functional features
- Responsive and mobile friendly
- jQuery plugin as a calendar
- Eventually runs on a http server

2.3 Assumptions

- Learning how to use jQuery plugin
- A Github repository to record commit history and access given to advisors
- The content is provided by the existing website
- Running the website with a http server

2.4 User Interaction and Design

Due to this project being a remake of an Existing website, the exact designs were implemented from Airbnb website's homepage.

2.5 Questions

2.5.1 How to Bind an event handler to the "scroll" JavaScript event, or trigger that event on an element?

This method is a shortcut for .on("scroll", handler).
The scroll event is sent to an element when the user scrolls to a different place in the element. It applies to window objects, but also to scrollable frames and elements with the overflow CSS property set to scroll (or auto when the element's explicit height or width is less than the height or width of its contents). [1]

2.5.2 How to choose a plugin and what should it do?

The first step of the calendar task was to find a plugin to use. after some research, we came to realization that 'DateRangePicker' was the perfect option, since it's dynamic and has a range to pick two different dates. [2]

2.5.3 How to add a dynamic calendar with a jQuery plugin?

To get started,it was necessary to include jQuery, Moment.js and Date Range Picker's files in the webpage. and the triggering the event by calling the plugin and finally customizing it.

2.6 Source Code

2.6.1 Adding a jQuery plugin to the project

A jQuery plugin is a method that we use to extend jQuery's prototype object. It is a piece of code written in a JavaScript file, which enables all jQuery objects to inherit any methods that you add. ... jQuery is a JavaScript library that allows web developers to add extra functionality to their websites. The pluggin that is used in this project is a plugin called 'DateRangePicker'. It's a dynamic calendar which allows two dates to get picked in a range. First, CDN scripts were added to the head of the HTML file.

```
  <script type="text/javascript"
      src="https://cdn.jsdelivr.net/jquery/latest/jquery.min.js"></script>
<script type="text/javascript"
    src="https://cdn.jsdelivr.net/momentjs/latest/moment.min.js"></script>
<script type="text/javascript"
    src="https://cdn.jsdelivr.net/npm/daterangepicker/daterangepicker.min.js"></script>
<link rel="stylesheet" type="text/css"
    href="https://cdn.jsdelivr.net/npm/daterangepicker/daterangepicker.css" />
```

Then to attach a date range picker to whatever you want to trigger it:

```
$('input[name="dates"]').daterangepicker();
```

And then it was time to customize Date Range Picker with options, and get notified when the user chooses new dates by providing a callback function. Also, some default features needed to get changed. like auto apply option.

```
  var DateRangePicker = function (element, options, cb) {
 this.autoApply = false;
 }
```

For changing the months to be appeared in their full name format this line of code was changed from monthShort() to month() [3]

```
 this.locale = {
 monthNames: moment.months(),
 }
```

[3]

2.6.2 Displaying a different navbar on Scroll

The .scroll method in jQuery helped us to be able to add event listeners to webpage when scroll happens.
This was in order to have a sticky navbar on scroll.

```
$(window).scroll(function () {
  const scroll = window.scrollY;
  if (scroll > 0) {
    $(".nav-main-scroll").removeClass("none").addClass("fixed");
    $(".nav-main").addClass("none");
    $(".nav-search-main").addClass("none");
    $(".nav-texts-scroll").addClass("none");
    $(".search-box-scroll").removeClass("none");
    $(".nav-search-main-scroll").addClass("none");
    $(".nav-main-scroll").removeClass("fixed-height");
  } else {
    $(".nav-main-scroll").addClass("none").removeClass("fixed");
    $(".nav-main").removeClass("none");
    $(".nav-search-main").removeClass("none");
    $(".nav-search-main-scroll").addClass("none");
    $(".nav-main-scroll").removeClass("fixed-height");
  }
 });
```

2.6.3 Styling the Calendar

For customizing the calendar, some changes needed to be applied in the daterangepicker.css file. styles like different hover effects, border radius, size and fonts.

```
.daterangepicker {
 position: absolute;
 color: inherit;
 background-color: #fff;
 width: 850px;
 padding: 0;
 margin-top: -90px;
 top: 100px;
 left: 20px;
 z-index: 3001;
 display: none;
 font-family: arial;
 font-size: 15px;
 line-height: 1em;
 border-radius: 32px;
}
```

```
.daterangepicker td.available:hover,
.daterangepicker th.available:hover {
 background-color: #eee;
 border-color: transparent;
 color: inherit;
 width: 48px;
 height: 47px;
}
```

```
.daterangepicker td.active,
.daterangepicker td.active:hover {
 background-color: rgb(34, 34, 34);
 border-color: transparent;
 color: #fff;
}
```

```
.daterangepicker .ranges li:hover {
 background-color: #eee;
}
```

2.6.4 Closing event for calendar

The calendar had a default option that would close the calendar, anytime you clicked uotside it. So to fix this problem and to prevent it getting closed inside it's container, we implemented this code :

```
outsideClick: function (e) {
      var target = $(e.target);
      // if the page is clicked anywhere except within the daterangerpicker/button
      // itself then call this.hide()
      if (
        // ie modal dialog fix
        e.type == "focusin" ||
        target.closest(this.element).length ||
        target.closest(this.container).length ||
        ($(e.target).closest(".flexible-main").length &&
          !$(e.target).closest(".flex-btn").length) ||
        target.closest(".calendar-table").length
      )
        return;
      this.hide();
      this.element.trigger("outsideClick.daterangepicker", this);
    }
```

The problem of the calendar position was also fixed by changing the default value and setting a specific one :

```
this.container.css({
        top: 0,
        left: 0,
        right: "auto",
      });
      var containerWidth = this.container.outerWidth();
      this.container.toggleClass("drop-up", drops == "up");
      if (this.opens == "left") {
        var containerRight =
          parentRightEdge -
          this.element.offset().left -
          this.element.outerWidth();
        if (containerWidth + containerRight > $(window).width()) {
          this.container.css({
            top: containerTop,
            right: 50,
            left: 9,
          });
        } else {
          this.container.css({
            top: containerTop,
            right: 50,
            left: "auto",
          });
        }
      } else if (this.opens == "center") {
        var containerLeft =
          this.element.offset().left -
          parentOffset.left +
          this.element.outerWidth() / 2 -
          containerWidth / 2;
        if (containerLeft < 0) {
          this.container.css({
            top: containerTop,
            right: 50,
            left: 9,
          });
        } else if (containerLeft + containerWidth > $(window).width()) {
          this.container.css({
            top: containerTop,
            left: "auto",
            right: 0,
          });
        } else {
          this.container.css({
            top: containerTop,
            left: containerLeft,
            right: 50,
          });
        }
      } else {
        var containerLeft = this.element.offset().left - parentOffset.left;
        if (containerLeft + containerWidth > $(window).width()) {
          this.container.css({
            top: containerTop,
            left: "auto",
            right: 50,
          });
        } else {
          this.container.css({
            top: 165,
            left: -5,
            right: 0,
          });
        }
      }
    }
```

3 Process Documentation

3.1 Plans, Estimates ,and Schedules

A 3 week time-line was considered for this project. The tasks were defined by/to Armin And Mobina. The first week, the HTML and CSS tasks were completed. The following week, JS tasks were in progress which were the most important and challenging part of this project. The final week was assigned to Responsive tasks, web server, Debugging and Refactoring. Also, the process was being documented the whole way from the beginning to the end.

3.2 Reports and Metrics

Everything on this project was a product of team-work. thus, tasks were defined fairly between both collaborators. The reports of accurate time spent on each task and the whole project is specified on 'Clockify'.

3.3 Standards

- Separate branches must be considered for each task
- The name of the branches must be defined according to their related task in the quire, following a definite rule including the information such as the name and number of the task and the person responsible for it
- All commit messages must be written according to the specified rules 2
- All the branches should be deleted at the end of the project, the only exciting branch should be 'Master' branch.

References

[1] JS Foundation - js.foundation, “.scroll() — jQuery API Documentation.”

[2] “Date Range Picker — JavaScript Date Time Picker Library.”

[3] “Show full month name in Daterange picker,” 03 2020.