**Name:** Komalpreet Kaur

**Student ID:** B01015647

**Assigned Tasks:** Implementing Booking section, My Booking, Footer Section respective HTML, CSS, and JavaScript components

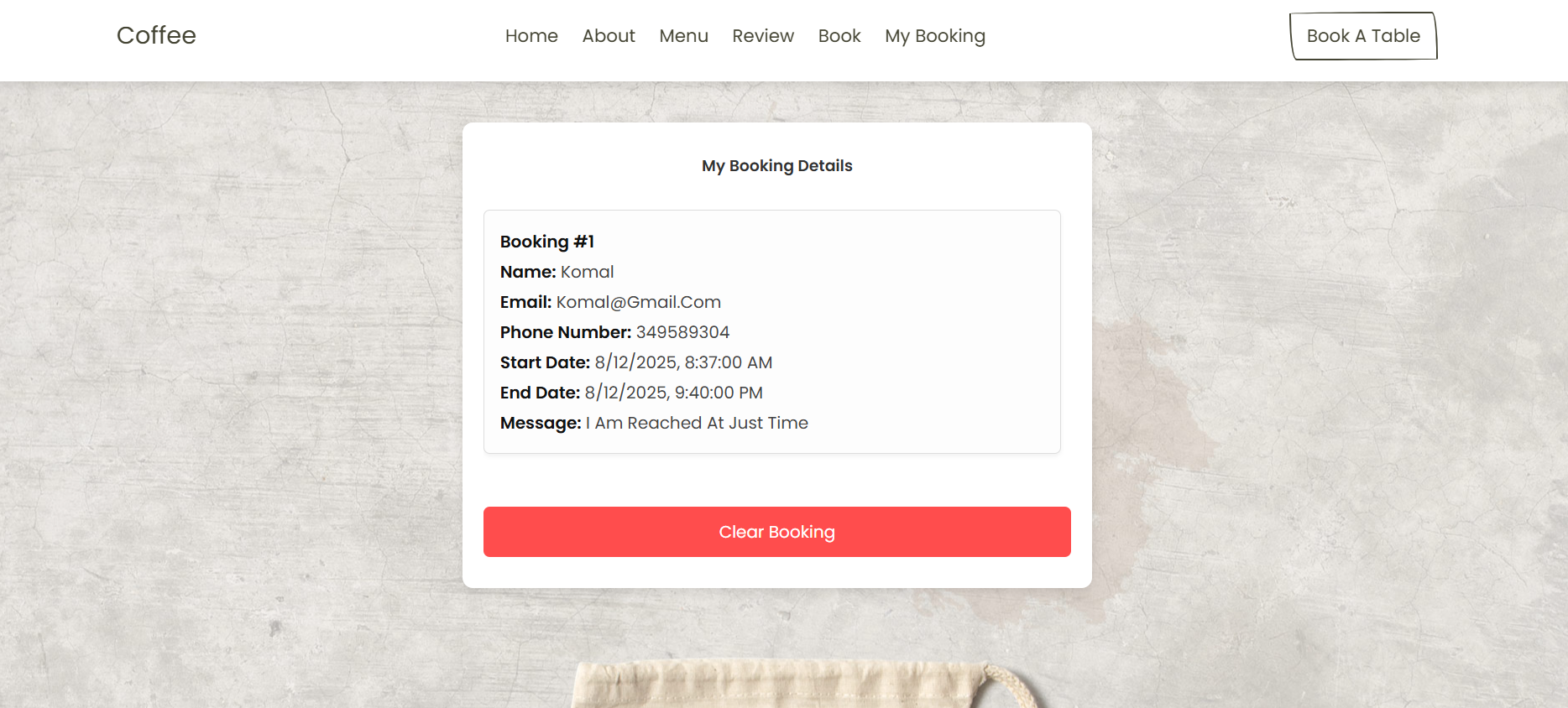
**Introduction**

I created the Booking Section, My Booking Page, and Footer Section as my own contribution to the collective project. I developed the booking system and informational footer using HTML, CSS, and JavaScript, which improved the website's usability and efficiency.

**Screenshot of the Booking a Table Section**



**Screenshot of the Show My booking**

****

**HTML Code Explanation of Booking Section**

I used and HTML5 doctype in my project so that the browser would render the code according to current standards. Next, I wrote to set the webpage's language to English. In the head area, I included a few crucial meta tags. For instance, I added to ensure that even older browsers use the most recent rendering engine, set the character set to Unicode with , and made the page responsive to mobile devices with . "Coffee" is the title I gave the page.

I linked some external files in the head. To begin, I inserted the favicon with . In order to make a responsive slider, I then included the Swiper library's CSS. In order to use the icons, I then included a link to Font Awesome. Last but not least, I included my own unique CSS file style.css, which manages the actual layout.

In the body, I first created a booking section with <section class="book" id="book">. Here, I used <h1 class="heading"> to give the title, while using <span> to style "booking" and "reserve a table" separately. Then, I created a <form> where users can input their information. I used required in the input boxes so that they cannot be left blank. I provided separate inputs for name, email, and number. I also provided datetime-local inputs to select the time and date, so that it is easy to set the reservation time. I also provided a textarea to write a message. Finally, I provided a submit button, whose value is "send message". I also provided a place to display a message to the user after the form submission with <div id="formMessage">.

In the <header> section, I have placed a navigation menu with links to Home, About, Menu, Reviews, Book and "My Booking". I have also added a button called book a table which takes the user to the booking section.

Then for the main content, I have used a div called .book which has a container. Inside it, I have given the title with an <h2> tag – My Booking Details. Then I have placed a <div id="bookingDisplay"> where the booking data will be displayed dynamically. Finally, I have placed a <button> which when pressed will delete all the booking data from the local storage.

**CSS Code Explanation of Booking Section**

To make the booking part visually appealing and easy to use, I first included a background image in the .book class of my CSS code. To maintain the focus in the center, I set the image's background to url(../image/book-bg.jpg) and its background-position to center. The background image is simultaneously spread around the entire area and attractively adjusts to various screen sizes when background-size:cover; is used. This gives the booking part a contemporary and polished visual effect.

I specifically styled the .book form. I used margin: 0 auto to center the form, so that it always appears in the middle. I set the maximum width to 60rem so that the form does not become unusually large on large screens and is easy for users to read. I gave sufficient padding: 3rem; inside the form so that there is enough space around the input box and buttons. I also used border-radius to make the corners of the form look soft and beautiful, and border: var(--border); to maintain theme-based consistency.

I made the input boxes (.book form.box) as easy to use as possible. To make the text easy to read, I kept the font size at 1.6 rem and utilized padding to give each box adequate space within. Each box takes up the entire width (width: 100%;). In order to modify the color of all inputs in accordance with the website theme, I grabbed the text color from a variable (var(--main-color)). To ensure that there is adequate space between each input field, I used margin:.7rem 0; in this instance. To make it clear to the user which field he is writing in, the border changes when the input is focused (.box:focus { border: var(--border-hover); }).

In the case of the text area, I gave a specific height (height: 15rem;) so that the layout of the form does not break even if a lot of writing is done. I also used resize: none; to prevent the user from manually changing the size of the text area, because doing so can make the entire layout messy.

In my myBooking.html file , I first reset the font, background color, and margins/padding for the body. I placed a background image in the .book class, and used Flexbox to center the content (horizontally and vertically).

I created a card-style in the .container class, using a white background, padding, border-radius, and shadow to make the content stand out. I set the font size and line height in the .booking-info class to make the text readable. I added margins, padding, borders, and a light shadow to the .booking-card to make the booking cards stand out.

Finally, I created a button with a red background for .btn-clear that gets darker on hover. I used media queries for responsive design, so that the booking info can be scrolled on small screens.

**JavaScript Code Explanation of Booking Section**

The JavaScript code for my booking part is structured such that every stage of the procedure is transparent. My primary objective was to get user information from the form and save it as an array in the browser's `localStorage`. Each reservation is saved as a distinct record in this way, and it can be shown as a card at a later time or all of the data can be removed with a single click. The key benefit is that the data is permanently retained, enhancing the user experience, even if the page is reloaded.

As soon as the form is submitted, I catch the event using `addEventListener('submit', ...)` and stop the page refresh with `event.preventDefault()`. Then I easily collect all the input values using the `FormData` object and organize them into a new object (`newData`) containing name, email, phone, date and message.

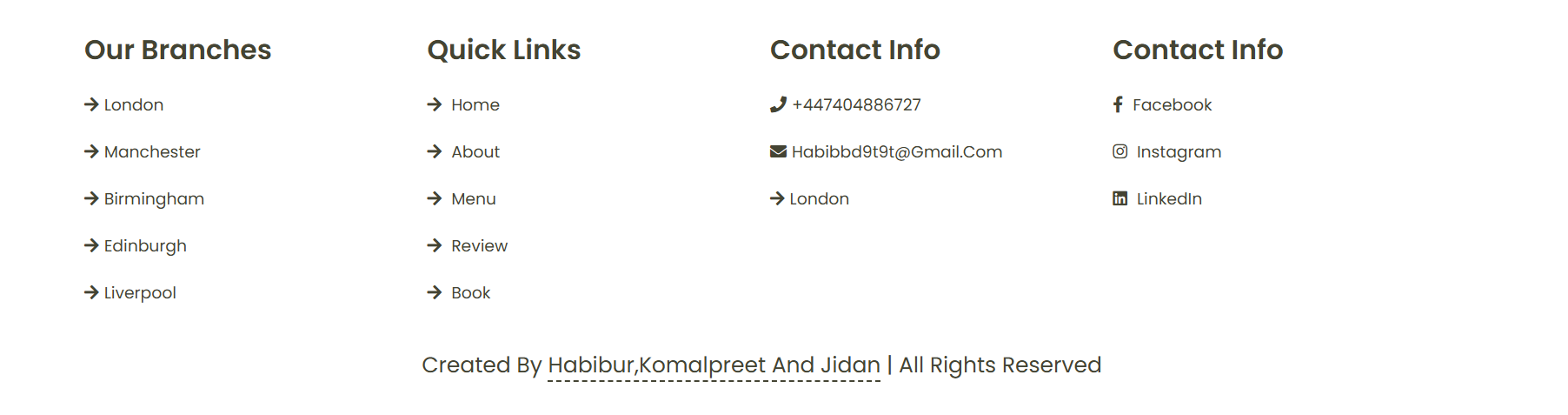
First, I checked if any data was previously saved. If there was data, I retrieved it as an array; if not, I used an empty array. Then I pushed the new booking data into that array and saved it using localStorage.setItem. Since localStorage can only store strings, I used JSON.stringify. I used console.log to make debugging easier, and finally I used form.reset() to clear the input fields so that the user could re-enter new data.

For displaying the data, I took the data from localStorage and created a separate card for each entry. Each card neatly displays the user name, email, phone number, start and end dates, and message. If there is no data, I display a message and hide the clear button. Once more, when the user selects the "clear" button, I execute "<localStorage.removeItem>" to eliminate all of the data and use "location.reload()`" to refresh the user interface.

Also, I have written custom logic to make the hash link work properly in the navbar. So that no matter what page the user is on, clicking on the nav link scrolls to the correct section.

Overall, my code streamlines the booking process—taking data from the form, saving it, displaying it in card form, and deleting it all together if needed. This makes the experience smoother for the user, the data is permanently stored, and my code is also easy to maintain or extend in the future.

**Screenshot of the Footer Section**



**HTML Code Explanation of Footer Section**

In the footer section, I used a few boxes. In the first box, I put the names of our branches (e.g. London, Manchester, etc.). In the second box, I have provided some quick links for quick navigation of the site. In the third box, I have provided contact info such as phone number, email and location. In another box, I have provided social media links (Facebook, Instagram, LinkedIn) so that visitors can follow our social profiles. I have used Font Awesome icons before each link to make it visually appealing. At the very bottom of the footer, I have placed a credit line where I have written the names of the team members and "all rights reserved".

I have included some scripts at the very end of the body. To make the slider function, I first supplied Swiper JS. After that, I added functionality and interaction using my own unique script.js.

Overall, using this HTML code, I have produced a comprehensive booking form, educational footer, and cookie banner. I have used custom CSS/JS in the code and other libraries to try to make it responsive, user-friendly, and professional.

**CSS Code Explanation of Footer Section**

I use .footer.box-container and organized it in a grid structure for the footer area. In this case, I utilized grid-template-columns and display: grid. In order to generate a layout with as many columns as the screen size will allow, repeat (auto-fit, minmax (23rem, 1fr)). To ensure there was adequate space between each box, I used gap: 1.5rem; This improves the footer's appearance and speeds up information retrieval for users.

To make them easily distinguishable, I have used the color from the main theme (var (--main-color)) and displayed the title (h3) of each box in the footer in a larger font (2.5 rem). To give you more room to click or touch, I made the links block level (display: block;). In order to create space between the icons and the text, I positioned icons next to the links and used padding-right. The icons shift to the right (padding-right: 2rem;) as the user hovers with the mouse, making for a pleasant visual interaction.

I've added a credits section to the footer's bottom. In this case, I've centered the text, used a high font size, and added enough padding. To give it a more polished appearance, I have highlighted each team member's name separately with border-bottom and positioned it inside the credits line in tags.

Lastly, to make the complete booking form and footer fit on various screen sizes, I utilized media queries. In order to keep the layout intact, I have lowered the font size to 55% and the form and footer grid when it falls below 991 pixels. To make it seem decent on mobile devices, I have centered all of the material in the footer, decreased the height of the text area, and further decreased the form's width when it falls below 768 pixels. In order to prevent the layout from shifting on small displays, I have also decreased the hover effect. I've made the form full screen, decreased the font size and padding, and consolidated all of the footer boxes into a single column when the screen size drops below 450 pixels.