Group members: Afshana Falza, Maroosha Ishtiaq, Timothy Dakis, Tanjinul Hoque

Gray Paper

Code Snippet #1

Snippet(s)

```
174
175   .contact-info > ul > li:nth-of-type(1) {
176       list-style-type: "";
177   }
178
179   .contact-info > ul > li:nth-of-type(2) {
180       list-style-type: "";
181   }
182
183   .contact-info > ul > li:nth-of-type(3) {
184       list-style-type: "";
185   }
```

Explanation

For the first 3 bullet points of the unordered list that is a child of the contact-info class, give them each a unique type of bullet point.

Code Snippet #2

Snippet(s)

```
      <!-- adapted from https://stackoverflow.com/questions/17169094/how-to-add-horizontal-line-in-a-table -->
      <tr class="table-division-line">
          <th colspan="3"></th>
      </tr>
```

```
7   /* adapted from https://stackoverflow.com/questions/17169094/how-to-add-horizontal-line-in-a-table */
8   .table-division-line > th {
9       border-top: 2px solid #333;
0   }
```

Explanation

Adds a row with an empty cell that spans the entire table that is then styled with CSS by making a solid border to draw a horizontal line within the table, similar to what an <hr> would do when used outside the table.

Code Snippet #3

Snippet(s)

```
// Search - taken / adapted from "50 Projects in 50 days" "Hidden Search Bar"
const search = document.querySelector(`.search`);
const btn = document.querySelector(`.btn`);
const input = document.querySelector(`.input`);

btn.addEventListener('click', () => {
    search.classList.toggle('active');
    input.focus();
})
```

Explanation

Acquires the elements that have the search, btn, and input classes respectively. Then, it checks if user clicks the element with the btn class. On click, it adds the active class to the element with the search class (so that a different set of CSS styles come into effect), then it puts focus on the element with the input class.

Code Snippet #4

Snippet(s)

```
56  .faq.active::before, .faq.active::after {
57      content: '\f075';
58      font-family: 'Font Awesome 5 Free';
59      color: lightgreen;
60      font-size: 7rem;
61      position: absolute;
62      opacity: 0.2;
63      top: 20px;
64      left: 20px;
65      z-index: 0;
66  }
67
68  .faq.active::before {
69      color: lightskyblue;
70      top: -10px;
71      left: -30px;
72      transform: rotateY(180deg);
73  }
```

Explanation

Uses ::before and ::after to add two child pseudo-elements to the FAQ box such that it adds 2 speech bubbles to the UI. It then uses fact CSS uses the latest defined property if specificity is the same to rotate one speech bubble, make it blue, and slightly move it. Overall this is to just improve the UI of the FAQ box.

Code Snippet #5

Snippet(s)

```
<div class="book-leave-rating" id="book-rating-3">Leave a Rating</div>
```

```
// Rating Button
for(let i=1; i<=5; i++) {
    let rate = document.querySelector(`#book-rating-${i}`);
    let ratebox = document.querySelector(`#rating-box-${i}`);

    rate.addEventListener('click', () => {
        ratebox.classList.remove("hidden");
    });
}
```

Explanation

Loops through all listed books (of which there are 5) and adds an event listener to each “Leave a Rating” div so that when the user clicks it, it removes the hidden class so that the CSS hiding the rating options for user is no longer effect and user can leave a rating.

Code Snippet #6

Snippet(s)

```
25  // Categories Button
26  let catbutton = document.querySelector(`#cat-container`);
27  let catArea = document.querySelector(`.categories`);
28  let clickCount = 0;
29  console.log(catArea);
30
31  catbutton.addEventListener('click', () => {
32      if(clickCount%2==0) {
33          console.log("clicked");
34          catArea.classList.remove("hidden");
35      }
36      else {
37          console.log("clicked");
38          catArea.classList.add("hidden");
39      }
40      clickCount++;
41  });
42
```

Explanation

Creates a counter and 2 variables such that when the user clicks the category menu icon, it checks how many times the user has pressed the menu so far. If prior to this click there was an even amount of clicks, the hidden class is removed from the form containing the categories so the CSS no longer hides it and the user can pick something, otherwise it adds hidden class again so the CSS hides it. In any case, the counter is incremented after.

Code Snippet #7

Snippet(s)

```
<!-- contact info -->
<div class="contact-info">
  <ul>
    <li>
      <strong>Call us</strong> at <a href="tel:999-917-book">999-917-book</a> for support, office hours are:
      <ul>
        <li>
          <table>
            <tr>
              <td><strong>Mon - Fri:</strong></td>
              <td>9am - 5pm EST</td>
            </tr>
            <tr>
              <td><strong>Sat - Sun:</strong></td>
              <td>1pm - 5pm EST</td>
            </tr>
          </table>
        </li>
      </ul>
    </li>
    <li>
      <strong>Email us</strong> at <a href="mailto:BookStore@support.com">BookStore@support.com</a> for 24-hour support
    </li>
    <li>
      <strong>Refer to</strong> the FAQ below for common questions
    </li>
  </ul>
</div>
```

Explanation

Creates an area that lists the contact info, as well as a line that says to check the FAQ. To achieve this an unordered list is used, with emphasis given to the first 2 words per bullets Additionally, the phone number bullet has a sub-bullet that creates a table that lists times users can call the office.

Code Snippet #8

Snippet(s)

```
    <!-- footer -->
    <div class="footer-basic">
      <footer>
          <ul class="list-inline">
              <li class="list-inline-item"><a href="https://github.com/afshanafalza" target="_blank">Afshana Falza</a></li>
              <li class="list-inline-item"><a href="https://github.com/mushi20" target="_blank">Maroosha Ishtiaq</a></li>
              <li class="list-inline-item"><a href="https://github.com/TimosDakis" target="_blank">Timothy Dakis</a></li>
              <li class="list-inline-item"><a href="https://github.com/Tanjinul" target="_blank">Tanjinul Hoque</a></li>
          </ul>
          <p class="copyright">BookStore &copy; 2023</p>
      </footer>
  </div>
```

Explanation

Creates a footer which lists all group members and links their GitHub page using absolute URLs.

Code Snippet #9

Snippet(s)

```
  <!-- navbar -->
  <nav class="nav">
    <div class="container">
      <a href="../index.html" class="current">BookStore</a>
      <ul>
        <li><a href="../index.html">Home</a></li>
        <li><a href="../bookView/book.html">Books</a></li>
        <li><a href="help.html">Help</a></li>
        <li><a href="../shoppingCart/shoppingCart.html"><i class="fas fa-shopping-cart"></i></a></li>
      </ul>
    </div>
</nav>
```

Explanation

Creates a nav bar which links to each web page of the website making use of relative URLs

Code Snippet #10

Snippet(s)

```
// taken / adapted from "50 Projects in 50 Days" "Scroll Animation"
const boxes = document.querySelectorAll('.faq')

window.addEventListener('scroll', checkBoxes)

checkBoxes()

function checkBoxes() {
    const triggerBottom = window.innerHeight / 5 * 4

    boxes.forEach(box => {
        const boxTop = box.getBoundingClientRect().top

        if(boxTop < triggerBottom) {
            box.classList.add('show')
        } else {
            box.classList.remove('show')
        }
    })
}
```

Explanation

Looks for all boxes that have the faq class and stores that in a variable. Also adds an event listener to the window itself that detects when the user scrolls and executes the checkBoxes function. The checkBoxes computes what 80% of the window's innerHeight it. It then loops through all boxes with the faq class, and obtains where the top of the box is located in the web page. If the top of the box is within the upper 80% of the window, the show class is added and CSS will apply styling to make the box appear, otherwise the show class is removed and the box will disappear. The checkBoxes function is also executed immediately as the web page is loaded so that any boxes in range show when the user loads the page.

Code Snippet #11

Snippet(s)

```
<!-- Carousel https://codepen.io/onion2k/pen/xxZYBVj -->
<section>
  <input type="radio" name="position" checked />
  <input type="radio" name="position" />
  <input type="radio" name="position" />
  <input type="radio" name="position" />
  <input type="radio" name="position" />
  <main id="carousel">
    <div class="item"><img src="./Book Images/Agatha Christie 1.jpg" alt="image-1"></div>
    <div class="item"><img src="./Book Images/Agatha Christie 2.jpg" alt="image-2"></div>
    <div class="item"><img src="./Book Images/Agatha Christie 3.jpg" alt="image-3"></div>
    <div class="item"><img src="./Book Images/Agatha Christie 4.jpg" alt="image-4"></div>
    <div class="item"><img src="./Book Images/Agatha Christie 5.jpg" alt="image-5"></div>
    <main>
</section>
```

Explanation

Creates a section, where in that section there are radio buttons and images, and through CSS these will be manipulated to create a carousel effect (refer to Code Snippets #12-16).

Code Snippet #12

Snippet(s)

```
47
48  /* Carousel https://codepen.io/onion2k/pen/xxZYBVj*/
49  section {
50    background-color: rgb(201, 228, 202);
51    height: 600px;
52    margin: 0;
53    display: grid;
54    grid-template-rows: 500px 100px;
55    grid-template-columns: 1fr 30px 30px 30px 30px 30px 1fr;
56    align-items: center;
57    justify-items: center;
58  }
```

Explanation

Creates a grid box with 2 rows and 7 columns, where the items are aligned and justified to the center.

Code Snippet #13

Snippet(s)

```
main#carousel {
  grid-row: 1 / 2;
  grid-column: 1 / 8;
  width: 50vw;
  height: 500px;
  display: flex;
  align-items: center;
  justify-content: center;
  overflow: hidden;
  transform-style: preserve-3d;
  perspective: 600px;
  --items: 5;
  --middle: 3;
  --position: 1;
  pointer-events: none;
}
```

Explanation

Sets it so the area containing the images of the carousel will span the entire first row of the grid box it is within. It also uses perspective to give it a 3d effect and uses transform-style to make the 3d effect persist with its children. Some variables are also set up. Any items that don’t fit within this area will not be displayed either.

Code Snippet #14

Snippet(s)

```css
img{
  width: 300px;
  height: 400px;
}

div.item {
  position: absolute;
  width: 300px;
  height: 400px;
  background-color: coral;
  --r: calc(var(--position) - var(--offset));
  --abs: max(calc(var(--r) * -1), var(--r));
  transition: all 0.25s linear;
  transform: rotateY(calc(-10deg * var(--r)))
    translateX(calc(-300px * var(--r)));
  z-index: calc((var(--position) - var(--abs)));
}
```

Explanation

Sets the image and the container the image is within to both have 300x400px dimensions. Then it declares some variables and does some computations to position the images. Effectively what the computations are ensuring is that for any items not selected there will be a slight tilt (using the fact it is 3d) either to the left or the right depending on the image's position from the one the user selects. Additionally, it ensures that for any image not selected it is moved by some interval of 300px to the left or the right (which results in only up to 3 images being displayed at a time, as any others get hidden). Finally, it ensures that for the item selected by the user, there is no tilt, and that it is put into the center of the main area (as translateX and rotateY will always be set to 0 for the selected item).

Code Snippet #15

Snippet(s)

```
div.item:nth-of-type(1) {
  --offset: 1;
  background-color: #90f1ef;
}
div.item:nth-of-type(2) {
  --offset: 2;
  background-color: #ff70a6;
}
div.item:nth-of-type(3) {
  --offset: 3;
  background-color: #ff9770;
}
div.item:nth-of-type(4) {
  --offset: 4;
  background-color: #ffd670;
}
div.item:nth-of-type(5) {
  --offset: 5;
  background-color: #e9ff70;
}
```

Explanation

For each item, have an offset variable that stores the offset to be used in the computation discussed in Code Snippet #14.

Code Snippet #16

Snippet(s)

```css
input:nth-of-type(1) {
  grid-column: 2 / 3;
  grid-row: 2 / 3;
}
input:nth-of-type(1):checked ~ main#carousel {
  --position: 1;
}

input:nth-of-type(2) {
  grid-column: 3 / 4;
  grid-row: 2 / 3;
}
input:nth-of-type(2):checked ~ main#carousel {
  --position: 2;
}

input:nth-of-type(3) {
  grid-column: 4 /5;
  grid-row: 2 / 3;
}
input:nth-of-type(3):checked ~ main#carousel {
  --position: 3;
}
```

```css
input:nth-of-type(4) {
  grid-column: 5 / 6;
  grid-row: 2 / 3;
}
input:nth-of-type(4):checked ~ main#carousel {
  --position: 4;
}

input:nth-of-type(5) {
  grid-column: 6 / 7;
  grid-row: 2 / 3;
}
input:nth-of-type(5):checked ~ main#carousel {
  --position: 5;
}
```

Explanation

For each radio button, make it take up one of the cells on the second row. Additionally, set the position variable based on the radio button that is checked such that it can be used in the computations discussed in Code Snippet #14 to ensure everything properly moved and tilted in the expected manner.

Code Snippet #17

Snippet(s)

```
// taken / adapted from "50 Projects in 50 Days" "FAQ-collapse"
const toggles = document.querySelectorAll('.faq-toggle')

toggles.forEach(toggle => {
    toggle.addEventListener('click', () => {
        toggle.parentNode.classList.toggle('active')
    })
})
```

Explanation

Creates a variable that stores every element with the faq-toggle class. It then loops through all of those elements and adds an event listener that checks if it gets clicked, and if it gets clicked it should add the active class to its parent.

Code Snippet #18

Snippet(s)

```
// taken / adapted from "50 Projects in 50 Days" and "Ripple Button"
const buttons = document.querySelectorAll('.ripple')

buttons.forEach(button => {
    button.addEventListener('click', function (e) {
        const x = e.pageX
        const y = e.pageY

        const buttonLeft = e.target.offsetLeft
        const buttonTop = e.target.offsetTop

        const parentLeft = e.target.parentElement.offsetLeft
        const parentTop = e.target.parentElement.offsetTop

        let xInside = x - parentLeft - buttonLeft
        let yInside = y - parentTop - buttonTop

        // if user presses the actual icon within button, find grandparent container's offsets to subtract to get correct position for ripple
        if(e.target.getAttribute("class") === "fas fa-chevron-down" || e.target.getAttribute("class") === "fas fa-times"){
            const grandparentLeft = e.target.parentElement.parentElement.offsetLeft
            const grandparentTop = e.target.parentElement.parentElement.offsetTop
            xInside -= grandparentLeft
            yInside -= grandparentTop
        }

        const circle = document.createElement('span')
        circle.classList.add('circle')
        circle.style.top = yInside + 'px'
        circle.style.left = xInside + 'px'

        this.appendChild(circle)

        setTimeout(() => circle.remove(), 500)
    })
})
```

Explanation

Adds ripple to button. First gets the position of where the click event occurred in the page. Then it computes the left and top positions of the element where the event occurred relative to its parent, and does the same with the parent of the element. Then it computes the x and y coordinates of where the circle will be by subtracting the differences of all 3 of these values (the circle is absolutely positioned as seen in Code Snippet #19, so subtracted off all distance that is not part of the button containing the circle to essentially create a separate coordinate plane within button for the circle).

Additionally, if user clicked the icon itself instead of the button surrounding the icon, then it also subtracts the grandparent's top and left positions relative to its parents (this is just because if the user clicks the icon itself, its parent becomes the surrounding button, but we still need the position of the surrounding button's parents like before).

It then creates a circle element (as a span), then adds the circle class to it, and then sets its CSS top and left properties to have the value of each computed value and finally appends the circle as a child. Then, after some time, the circle is removed from the DOM.

Code Snippet #19

Snippet(s)

```css
/* Ripple button styling - taken / adapted "50 Projects in 50 Days" "Ripple Button" */
button .circle {
    position: absolute;
    background-color: purple;
    width: 10px;
    height: 10px;
    border-radius: 50%;
    transform: translate(-50%, -50%) scale(0);
    animation: scale 0.5s ease-out;
  }

@keyframes scale {
    to {
      transform: translate(-50%, -50%) scale(3);
      opacity: 0;
    }
  }
```

Explanation

Styles any descendants of a button with the circle class. It then applies some transformations, namely so that the circle can scale up using an animation. The animation is defined such that the circle will get bigger and fade out.

Code Snippet #20

Snippet(s)

```
186
187   ul ul {
188       list-style-type: none;
189       padding-left: 0.5em;
190   }
191
```

Explanation

When there are nested unordered lists, the second bullet point does not display a bullet point icon, but still has padding to indent it.

Code Snippet #21

Snippet(s)

```css
/* Top button styling */
.top-button > button{
    margin-bottom: 30px;
    padding: 20px 40px;
    border-radius: 10px;
    background-color: var(--box-color);
    border: 1px solid #9fa4a8;
    cursor: pointer;
    font-size: 20px;
    font-weight: bold;
    transition: background-color 0.15s ease-in;
}

.top-button > button:hover, .top-button > button:focus {
    transition: background-color 0.15s ease-in;
    background-color: var(--box-hover-color);
}
```

Explanation

Styles the button that is a child of the top-button class. Creates a rounded box and makes the cursor display as a pointer. Then it has a transition on the background color as when the user hovers or focuses on the button the background color changes, and the use of transitions makes that color change more smooth.

Code Snippet #22

Snippet(s)

```
<!-- book display -->
<div id="book-display">
    <div id="book-row">
        <!-- book 1 -->
        <div class="book-info" id="book-info-1">
            <div class="book-area">
                <img class="book-cover" src="bookcovers\bookcover1.jpg" height="360.6">
            </div>
            <div class="book-under">
                <div class="book-details">
                    <div class="book-title">
                        And There He Kept Her
                    </div>
                    <div class="book-author">
                        by <b>Joshua Moehling</b>
                    </div>
                    <div class="book-price">
                        $17.99
                    </div>
                </div>
                <div class="rating-area">
                    <div class="stars">
                        <span class="material-icons md-18 gold review-button">star</span>
                        <span class="material-icons md-18 gold review-button">star</span>
                        <span class="material-icons md-18 gold review-button">star</span>
                        <span class="material-icons md-18 gold review-button">star</span>
                        <span class="material-icons md-18 gold review-button">star</span>
                    </div>
                    <div class="book-rating">(348)</div>
                </div>
                <!-- leave a rating -->
                <div class="book-leave-rating" id="book-rating-1">Leave a Rating</div>
                <div class="rating-box hidden" id="rating-box-1">
                    <form>
                        <input type="radio" id="html" name="fav_language" value="HTML">
                        <label for="html"><span class="material-icons md-18 gold review-button">star</span><span class="material-icons md-18 gold review-button">star</spa
                        <input type="radio" id="html" name="fav_language" value="HTML">
```

Explanation

Creates a division to display books, and within that it creates a division for a row of books, and within that it adds a division for each book. Each book d splays the title, author, price, and ratings (where stars are generated using CSS). Then, there is also a leave a rating area that uses a form of radio buttons, and each has a label consisting of the amount of stars the user can select (where stars are generated using CSS).

Code Snippet #23

Snippet(s)

```
// taken from "50 Projects in 50 Days" "Sticky Nav Bar"
const nav = document.querySelector('.nav')
window.addEventListener('scroll',fixNav)

function fixNav(){
  if(window.scrollY > nav.offsetHeight + 150){
    nav.classList.add('active')
  } else {
    nav.classList.remove('active')
  }
}
```

Explanation

Stores the nav element in a variable. Then adds an event listener to the window, such that every time the user scrolls the fixNav function fires. The fixNav function checks if the amount the user has scrolled is more than the overall height of the navbar + 150. If the condition is true, the active class is added to the navbar and thus the CSS properties associated with it are applied, otherwise the active class is removed and so similarly the CSS properties associated with it are removed.

Code Snippet #24

Snippet(s)

```
<!-- total cost -->
<tr>
    <td class="total">Total</td>
    <td class="total"></td>
    <td class="total">$105</td>
</tr>
<tr>
    <td class="total">Shipping Estimate</td>
    <td class="total"></td>
    <td class="total">Calculated at Checkout</td>
</tr>
<tr>
    <th class="total">Order Total</th>
    <td class="total"></td>
    <td class="total"><strong>$105</strong></td>
</tr>
```

```
.total:nth-child(1){
    text-align: left;
}

.total:nth-child(3){
    text-align: right;
}
```

Explanation

In the HTML certain th and td's are given a total class, such that through the CSS we can align the text to either the left or the right depending on which child of the overall tr it is.

Code Snippet #25

Snippet(s)

```
<div class="faq">
  <h3 class="faq-title">
    How do I get a refund on my order?
  </h3>
  <p class="faq-text">
    Please email us with your order number within 7 days of purchase.
  </p>
  <button class="faq-toggle ripple">
    <i class="fas fa-chevron-down"></i>
    <i class="fas fa-times"></i>
  </button>
</div>
```

Explanation

Create an overall FAQ container, and within that there is a header for the question and a paragraph for the answer. Additionally, there is a button for the user to press to toggle it, which contains two classes which will ultimately display two different Font Awesome icons (which one is displayed is decided by CSS).

Code Snippet #26

Snippet(s)

```
</head>
<body id="top">
```

```
<!-- top button -->
<a href="#top" class="top-button">
  <button>
    Top
  </button>
</a>
```

Explanation

Puts an id of “top” in the body tag, so that it can be linked within the anchor so that the button, when pressed, links to the top of the webpage.

Code Snippet #27

Snippet(s)

```css
.material-icons.light-grey { color: rgba(0, 0, 0, 0.259); }

.material-icons.dark-grey { color: rgba(0, 0, 0, 0.68); }

.material-icons.md-18 { font-size: 18px }

.material-icons.gold { color:rgba(214, 178, 32, 0.711)}
```

Explanation

This CSS snippet is used to style the stars used for reviews such that there can be gold star as well as gray stars. It also sets the size of all the stars.

Code Snippet #28

Snippet(s)

```css
.search .input {
    background-color: #fff;
    border: 0;
    font-size: 18px;
    padding: 15px;
    height: 15px;
    width: 15px;
    position: relative;
    right: 0px;
    transition: width 0.3s ease;
}

.btn {
    background-color: rgb(201, 228, 202);
    border: 0;
    cursor: pointer;
    font-size: 24px;
    position: absolute;
    top: 0;
    left: 0;
    height: 50px;
    width: 50px;
    transition: transform 0.3s ease;
}
```

```css
.search.active .input {
    width: 200px;
}

.search.active .btn {
    transform: translateX(198px);
}
```

Explanation

This snippet styles the search input field and search icon. What it also does is that when through a JS script (discussed in Code Snippet #3) the active class is applied to the search container, the element with the input class gets wider and the element with the btn class gets moved, where transitions are used to make this smoother.

Code Snippet #29

Snippet(s)

```
.faq:nth-of-type(even) {
    transform: translateX(-400%);
}
```

Explanation

Every even element with the FAQ class will be moved a significant amount to the left such that the user can no longer see it.

Code Snippet #30

Snippet(s)

```
.faq-toggle .fa-times {
    display: none;
}

.faq.active .faq-toggle .fa-times {
    color: #fff;
    display: block;
}
```

Explanation

Uses the way CSS applies styles in a way such that if the element with the faq class does not also have the active class, the element with the fa-times class is not displayed, but when the element with the faq class gets the active class, then the element with the fa-times class is displayed as a block.