The Chop Shop – Gray Paper

<head> Provides meta data which includes information such as the <title> (title of website in the tab of the browser. Also contains the <link> for external sources such as icons and style sheets.

```html
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width", inital-scale="1.0">
        <link rel="stylesheet" href="index_styles.css">
        <title>The Chop Shop | Home</title>
        <link rel="icon" type="image/png" href="images/rusty_truck.png">
        <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.1/css/all.min.css"
    </head>
```

Within the <body>, the <nav> tag contains the sticky navigation which is anchored at the top of each page. Uses <ul> and <li> tags to create a list of links to each page.

```html
<body>
    <nav class="nav navbg">
        <div class="container">
            <h1 class="logo"><a href="index.html"><img src="images/Chop_shop_logo.png" alt="Chop Shop Logo"></a></h1>
            <ul class="list-text">
                <li><a href="index.html" class="current">home</a></li>
                <span>|</span>
                <li><a href="services.html">services</a></li>
                <span>|</span>
                <li><a href="about.html">about</a></li>
                <span>|</span>
                <li><a href="gallery.html">gallery</a></li>
                <span>|</span>
                <li><a href="contact.html">contact</a></li>
            </ul>
        </div>
    </nav>
```

Style sheet for the navigation bar uses the classes contained within each <div> section to access specific elements in order to style the navigation bar and its elements. Uses various properties to anchor the navigation bar at the top of the page (position: fixed), making it appear on top (z-index: 1) of other elements. Styling properties such as "background," "color," and "text-shadow" create an aesthetic appropriate for the business. Transition properties and the :active properties allow the user to interact with various elements as they appear to move.

The background properties utilizes the linear-gradient() option along with a url image to become the navigation bar background where the alpha channel adjusts the opacity of the image.

```css
.nav {
    position: fixed;
    background-color: ◼gray;
    top: 0;
    left: 0;
    right: 0;
    transition: all .3s ease-in-out;
    border-radius: 0px 0px 15px 15px;
    box-shadow: 0 8px 10px ◼gray;
    z-index: 1;
}
.navbg {
    background: linear-gradient(
        □rgba(0, 0, 0, 0.5),
        □rgba(0, 0, 0, 0.5)),
        url(images/Nav_bg.jpg) no-repeat center center/cover;

}

.nav .container {
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: 20px;
    transition: all .3s ease-in-out;
}

.nav ul {
    display: flex;
    list-style-type: none;
    gap: 5px;
    align-items: center;
    justify-content: center;
}
```

JavaScript code for the navigation bar uses an active listener to register the point at which the scroll height is greater than the offset height. This allows the fixNav function to set an active class to the transition properties within the style sheet to make the navigation bar scale its appearance smaller when scrolling down the page.

```javascript
// navbar
const nav = document.querySelector('.nav')
window.addEventListener('scroll', fixNav)

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

JavasCript code for the homepage expanding panel. Selects the panel class and sets the active class for the transition property. The active listener registers when a panel is "clicked." The active class is added to the transition properties that creates an expanding affect for each panel selected.

```javascript
// panel
const panels = document.querySelectorAll('.panel')

console.log(panels[1])

panels.forEach((panel) => {
    panel.addEventListener('click', () => {
        removeActiveClasses()
        panel.classList.add('active')
    })
})

function removeActiveClasses() {
    panels.forEach(panel => {
        panel.classList.remove('active')
    })
}
```

The panel class in the style sheet sets the border, background, border-radius, margin properties, etc. to make each panel appear uniform and appear side by side (display: flex)

```css
.panel {
    border: 1px solid █white;
    background-size: auto 100%;
    background-position: center;
    background-image: center;
    background-repeat: no-repeat;
    height:65vh;
    border-radius: 30px;
    color: █white;
    cursor:default;
    flex: 1px;
    margin-top:18em;
    margin-right: 1.5em;
    margin-left: 1.5em;
    margin-bottom: 0;
    transition: flex 0.5s ease-in;
}
```

The Services page uses a table to contain the services. Using the table rows and table data to correspond the service to an image.

```html
<div class="panel" style="background-image: url(images/Oil_service.jpg);">
    <h3>lube express</h3>
    <section class="services-container">
        <caption><h2>Services</h2></caption>
        <table class="brake-table">
            <tbody>
                <tr>
                    <td><img src="images/hand_holding_wrench_front.png" width="125px" height="50px"></td>
                    <td>Oil Change</td>
                </tr>
                <tr>
                    <td><img src="images/hand_holding_wrench_front.png" width="125px" height="50px"></td>
                    <td>Chassis Lubrication</td>
                </tr>
                <tr>
                    <td><img src="images/hand_holding_wrench_front.png" width="125px" height="50px"></td>
                    <td>Differential Service</td>
                </tr>
                <tr>
                    <td><img src="images/hand_holding_wrench_front.png" width="125px" height="50px"></td>
                    <td>Transmission Fluid Change</td>
                </tr>
                <tr>
                    <td><img src="images/hand_holding_wrench_front.png "width="125px" height="50px"></td>
                    <td>Greasing Joints & Bearings</td>
                </tr>
            </tbody>
        </table>
    </section>
</div>
```

Active and hover control what actions take place when the cursor either hovers over the object or actively click the object. Within the stylesheet, we're able to create transition affects to scale the container, or make the color of the object change when hovering or clicking.

```css
.button button:hover {
  background-color: ▢rgba(201, 201, 201, 0.463);
  color: ▢white;
  transform: scale(1.2);
  transition: 0.3s ease-in-out;
}

.button button:active {
  background-color: ▢#5c3030;
  transform: scale(.99);
  transition: 0.2s;
}
```

JavaScript code creates a slideshow affect for the testimonials. User information is stored in an array, where a function assigns each value to the objects within the html. A set interval function is used to update each stored information contained within the array, to the html testimonial section with the corresponding object names.

```javascript
const testimonials = [
  {
    name: 'Kanye South',
    job: 'Hamster Trainer',
    photo: 'https://randomuser.me/api/portraits/men/43.jpg',
    text: "My experience at The Chop Shop was nothing short of exceptional. From the warm welcome at the doo
  },
  {
    name: 'Beyonce Coles',
    job: 'Body Builder',
    photo: 'https://randomuser.me/api/portraits/women/43.jpg',
    text: "I couldn't be happier with the outstanding service I received at The Chop Shop! From the moment
  },
  {
    name: 'Aubrey Graham',
    job: 'Unemployed',
    photo: 'https://randomuser.me/api/portraits/men/21.jpg',
    text: "My visit to The Chop Shop was a complete disappointment. The customer service was lackluster, and
  },
  {
    name: 'Vick Butts',
    job: 'Cashier',
    photo: 'https://randomuser.me/api/portraits/men/43.jpg',
    text: "My experience at The Chop Shop was okay. The staff was friendly enough, and the service on my veh
  },
  {
    name: 'Mary B. Poppin',
    job: 'Nurse',
    photo: 'https://randomuser.me/api/portraits/women/37.jpg',
    text: "I recently took my car to The Chop Shop for some routine maintenance and was met with a mixed exp
  },
]

let idx = 1;

function updateTestimonial() {
  const {name, job, photo, text} = testimonials[idx]

  testimony.innerHTML = text
  userImage.src = photo
  username.innerHTML = name
  occupation.innerHTML = job

  idx++

  if(idx > testimonials.length - 1) {
    idx = 0
  }
}

setInterval(updateTestimonial, 10000)
```

<form> tags used within the Service and Contact pages uses a POST method attribute that delivers the information entered within the body and not in the url (GET) for security purposes. The forms uses input type attributes such as "text" for names, "email" to ensure the correct format for an email address is used, radio buttons where the user can select a single option of the ones presented, checkboxes to select various options, text-area to type personalized messages, select attributes to create a drop-down menu to select options, etc.

```html
<section class="contact">
    <div class="container">
        <h2>Contact Us</h2>
        <div class="contact-container">
            <div class="contact-form">
                <h3>Send us a message</h3>
                <form action="" method="post">
                    <div class="form-container">
                        <input type="text" name="firstName" placeholder="Enter first name">
                    </div>
                    <div class="form-container">
                        <input type="text" name="Lastname" placeholder="Enter last name">
                    </div>
                    <div class="form-container">
                        <input type="tel" id="phone" name="phone" pattern="[0-9]{3}-[0-9]{3}-[0-9]{4}" placeholder="Format: 123-456-7890" required>
                    </div>
                    <div class="form-container">
                        <input type="email" id="email" name="email" placeholder="username@domain.com" required>
                    </div>
                    <div class="method-container">
                        <label>Method of Contact:</label>
                        <input type="checkbox" id="emailContact" name="contactMethod" value="email">
                        <label for="emailContact">Email</label>
                        <input type="checkbox" id="callContact" name="contactMethod" value="call">
                        <label for="phoneContact">Call</label>
                        <input type="checkbox" id="texContact" name="contactMethod" value="text">
                        <label for="textContact">Text</label>
                    </div>
```