**SMART PUBLIC RESTROOM**

**HTML CODE:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta http-equiv="X-UA-Compatible" content="IE=edge" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>Smart Public Restroom</title>

<!--<style> body{

background-image:url('bg.webp');

background-repeat: no-repeat;

background-attachment: fixed;

background-size: 100% 100%;}

</style>

<!-- custom css -->

<link rel="stylesheet" href="./styles/styles.css" />

<!-- CSS only for bootstrap -->

<link

href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css"

rel="stylesheet"

integrity="sha384-rbsA2VBKQhggwzxH7pPCaAqO46MgnOM80zW1RWuH61DGLwZJEdK2Kadq2F9CUG65"

crossorigin="anonymous"

/>

<!-- font-awesome icons -->

<link

rel="stylesheet"

href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.2.1/css/all.min.css"

integrity="sha512-MV7K8+y+gLIBoVD59lQIYicR65iaqukzvf/nwasF0nqhPay5w/9lJmVM2hMDcnK1OnMGCdVK+iQrJ7lzPJQd1w=="

crossorigin="anonymous"

referrerpolicy="no-referrer"

/>

</head>

<body onload="loadInitialData()">

<!-- Btn which should be displayed on smaller screen -->

<div class="add\_newbtnmobile\_only d-md-none">

<button

class="btn btn-primary d-flex align-items-center gap-2"

type="submit"

data-bs-toggle="modal"

data-bs-target="#addNewModal"

>

<i class="fas fa-plus"></i>Search here

</button>

</div>

<!-- too add new modal on click of add an item btn -->

<div

class="modal fade"

id="addNewModal"

tabindex="-1"

aria-labelledby="addNewModalLabel"

aria-hidden="true"

>

<div class="modal-dialog modal-dialog-centered">

<div class="modal-content">

<div class="modal-header">

<h5 class="modal-title" id="addNewModalLabel">Search here</h5>

<button

type="button"

class="btn-close"

data-bs-dismiss="modal"

aria-label="Close"

></button>

</div>

<div class="modal-body">

<form onsubmit="return false">

<div class="mb-3">

<label for="imageUrl" class="form-label">Image URL</label>

<input

type="url"

class="form-control"

id="imageUrl"

placeholder="http://example.com/image.png"

/>

</div>

<div class="mb-3">

<label for="taskTitle" class="form-label">Near by</label>

<input

type="text"

class="form-control"

id="taskTitle"

placeholder="Enter a land mark"

required

/>

</div>

<div class="mb-3">

<label for="tags" class="form-label">Location</label>

<input

type="text"

class="form-control"

id="tags"

placeholder="Enter your location here"

required

/>

</div>

<div class="mb-3">

<label for="taskDescription" class="form-label"

>Task Description</label

>

<textarea

class="form-control"

id="taskDescription"

placeholder="Explain something about it..."

rows="4"

required

></textarea>

</div>

<button

type="button"

class="btn btn-secondary"

data-bs-dismiss="modal"

>

Close

</button>

<button

type="submit"

class="btn btn-primary"

onclick="handleSubmit()"

>

Save changes

</button>

</form>

</div>

</div>

</div>

</div>

<!-- Modal for UI Display On Home pg -->

<div

class="modal fade"

id="showTask"

tabindex="-1"

aria-labelledby="showTaskLabel"

aria-hidden="true"

>

<div class="modal-dialog modal-lg">

<div class="modal-content">

<div class="modal-header">

<h1 class="modal-title" id="showTaskLabel">Task Details</h1>

<button

type="button"

class="btn-close"

data-bs-dismiss="modal"

aria-label="Close"

></button>

</div>

<div class="modal-body task\_modal\_body">

<!-- <p>js txt</p> -->

</div>

<div class="modal-footer">

<button

type="button"

class="btn btn-secondary"

data-bs-dismiss="modal"

>

Close

</button>

</div>

</div>

</div>

</div>

<!-- Nav bar -->

<nav class="navbar navbar-expand-md bg-light">

<div class="container-fluid">

<a class="navbar-brand" href="#">Tasky</a>

<button

class="navbar-toggler"

type="button"

data-bs-toggle="collapse"

data-bs-target="#navbarSupportedContent"

aria-controls="navbarSupportedContent"

aria-expanded="false"

aria-label="Toggle navigation"

>

<span class="navbar-toggler-icon"></span>

</button>

<div class="collapse navbar-collapse" id="navbarSupportedContent">

<ul class="navbar-nav me-auto mb-2 mb-lg-0">

<li class="nav-item">

<a class="nav-link active" aria-current="page" href="#">Home</a>

</li>

</ul>

<!-- <form class="d-flex" role="search"> -->

<button

class="btn btn-primary d-flex align-items-center gap-2"

type="submit"

data-bs-toggle="modal"

data-bs-target="#addNewModal"

>

<i class="fas fa-plus"></i>

Add New Task

</button>

<!-- </form> -->

</div>

</div>

</nav>

<!-- entire body - search i/p with display cards -->

<div class="container">

<!-- Content here -->

<section class="mt-4">

<div class="row justify-content-center">

<div class="col-md-6">

<div class="input-group flex-nowrap shadow-lg rounded">

<input

type="search"

class="form-control"

placeholder="Search Task"

aria-label="Search Task"

aria-describedby="addon-wrapping"

oninput="searchTask.apply(this, arguments)"

/>

<span class="input-group-text" id="addon-wrapper">

<i class="fas fa-search"></i>

</span>

</div>

</div>

</div>

</section>

<section class="mt-4">

<div class="row task\_\_contents">

<!-- <h1>This is the place where we will get all the cards</h1> -->

</div>

</section>

</div>

<!-- scripts -->

<script type="text/javascript" src="./scripts/index.js"></script>

<script

src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.min.js"

integrity="sha384-oBqDVmMz9ATKxIep9tiCxS/Z9fNfEXiDAYTujMAeBAsjFuCZSmKbSSUnQlmh/jp3"

crossorigin="anonymous"

></script>

<script

src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/js/bootstrap.min.js"

integrity="sha384-cuYeSxntonz0PPNlHhBs68uyIAVpIIOZZ5JqeqvYYIcEL727kskC66kF92t6Xl2V"

crossorigin="anonymous"

></script>

</body>

</html>

**CSS CODE:**

\*{

padding: 0;

margin: 0;

}

body{

background-color: powderblue !important;

}

.add\_newbtnmobile\_only{

position: fixed;

bottom: 0;

left: 0;

width: 100%;

background: white;

padding: 15px 10px;

z-index: 3;

}

/\*

@media (min-width: 780px){

.add\_newbtnmobile\_only{

display: none;

}

} \*/

**JAVASCRIPT CODE:**

const state = {

taskList: [],

};

// dom manipulations

const taskModal = document.querySelector(".task\_modal\_body");

const taskContents = document.querySelector(".task\_\_contents");

// to create a card on home pg

const htmlTaskContent = ({ id, title, description, type, url }) => `

<div class='col-md-6 col-lg-4 mt-3' id=${id} key=${id}>

<div class='card shadow-sm task\_\_card'>

<div class='card-header d-flex gap-2 justify-content-end task\_card\_header'>

<button type='button' class='btn btn-outline-info mr-2' name=${id} onclick="editTask.apply(this, arguments)">

<i class='fas fa-pencil-alt' name=${id}></i>

</button>

<button type='button' class='btn btn-outline-danger mr-2' name=${id} onclick="deleteTask.apply(this, arguments)">

<i class='fas fa-trash-alt' name=${id}></i>

</button>

</div>

<div class='card-body'>

${

url

? <img width='100%' height='150px' style="object-fit: cover; object-position: center" src=${url} alt='card image cap' class='card-image-top md-3 rounded-lg' />

: <img width='100%' height='150px' style="object-fit: cover; object-position: center" src="https://tse3.mm.bing.net/th?id=OIP.LZsJaVHEsECjt\_hv1KrtbAHaHa&pid=Api&P=0" alt='card image cap' class='card-image-top md-3 rounded-lg' />

}

<h4 class='task\_card\_title'>${title}</h4>

<p class='description trim-3-lines text-muted' data-gram\_editor='false'>

${description}

</p>

<div class='tags text-white d-flex flex-wrap'>

<span class='badge bg-primary m-1'>${type}</span>

</div>

</div>

<div class='card-footer'>

<button

type='button'

class='btn btn-outline-primary float-right'

data-bs-toggle='modal'

data-bs-target='#showTask'

id=${id}

onclick='openTask.apply(this, arguments)'>

Open Task

</button>

</div>

</div>

</div>

`;

// Dynamic modals(cards) on our home page/ui

const htmlModalContent = ({ id, title, description, url }) => {

const date = new Date(parseInt(id));

return `

<div id = ${id}>

${

url

? <img width='100%' src=${url} alt='card image here class='img-fluid place\_\_holder\_\_image mb-3'/>

: <img width='100%' height='150px' style="object-fit: cover; object-position: center" src="https://tse3.mm.bing.net/th?id=OIP.LZsJaVHEsECjt\_hv1KrtbAHaHa&pid=Api&P=0" alt='card image cap' class='card-image-top md-3 rounded-lg' />

}

<strong class="text-sm text-muted">Created on ${date.toDateString()}</strong>

<h2 class="my-3">${title}</h2>

<p class='lead'>${description}</p>

</div>`;

};

// here we will be updating our local storage (i.e., the modals/cards which we see on our ui)

const updateLocalStorage = () => {

localStorage.setItem(

"task",

JSON.stringify({

tasks: state.taskList,

})

);

};

// to get data or card or modals on ur ui from local storage (Browsers storage)

const loadInitialData = () => {

const localStorageCopy = JSON.parse(localStorage.task);

if (localStorageCopy) state.taskList = localStorageCopy.tasks;

state.taskList.map((cardDate) => {

taskContents.insertAdjacentHTML("beforeend", htmlTaskContent(cardDate));

});

};

const handleSubmit = (event) => {

const id = ${Date.now()};

const input = {

url: document.getElementById("imageUrl").value,

title: document.getElementById("taskTitle").value,

type: document.getElementById("tags").value,

description: document.getElementById("taskDescription").value,

};

if (input.title === "" || input.type === "" || input.description === "") {

return alert("Please Fill All The Fields");

}

taskContents.insertAdjacentHTML(

"beforeend",

htmlTaskContent({

...input,

id,

})

);

// updated task list - for 1st go

state.taskList.push({ ...input, id });

// update the same on localStorage too

updateLocalStorage();

};

// opens new modal on our ui when user clicks open task

const openTask = (e) => {

// pop up the current one

if (!e) e = window.event;

// find the crt card opened

const getTask = state.taskList.find(({ id }) => id === e.target.id);

taskModal.innerHTML = htmlModalContent(getTask);

// console.log(getTask);

};

// delete operation

const deleteTask = (e) => {

if (!e) e = window.event;

const targetID = e.target.getAttribute("name");

// console.log(targetID);

const type = e.target.tagName;

// console.log(type)

const removeTask = state.taskList.filter(({ id }) => id !== targetID);

// console.log(removeTask);

state.taskList = removeTask;

updateLocalStorage();

if (type === "BUTTON") {

// console.log(e.target.parentNode.parentNode.parentNode);

return e.target.parentNode.parentNode.parentNode.parentNode.removeChild(

e.parentNode.parentNode.parentNode

);

}

// console.log(e.target.parentNode.parentNode.parentNode.parentNode);

return e.target.parentNode.parentNode.parentNode.parentNode.parentNode.removeChild(

e.target.parentNode.parentNode.parentNode.parentNode

);

};

// edit operation

const editTask = (e) => {

if (!e) e = window.event;

const targetID = e.target.id;

const type = e.target.tagName;

let parentNode;

let taskList;

let taskDescription;

let taskType;

let submitButton;

if (type === "BUTTON") {

parentNode = e.target.parentNode.parentNode;

} else {

parentNode = e.target.parentNode.parentNode.parentNode;

}

taskTitle = parentNode.childNodes[3].childNodes[3];

taskDescription = parentNode.childNodes[3].childNodes[5];

taskType = parentNode.childNodes[3].childNodes[7].childNodes[1];

submitButton = parentNode.childNodes[5].childNodes[1];

// console.log(taskTitle);

taskTitle.setAttribute("contenteditable", "true");

taskDescription.setAttribute("contenteditable", "true");

taskType.setAttribute("contenteditable", "true");

// needs to be implemented

submitButton.setAttribute("onclick", "saveEdit.apply(this, arguments)");

submitButton.removeAttribute("data-bs-toggle");

submitButton.removeAttribute("data-bs-target");

submitButton.innerHTML = "Save Changes";

};

const saveEdit = (e) => {

if (!e) e = window.event;

const targetID = e.target.id;

const parentNode = e.target.parentNode.parentNode;

const taskTitle = parentNode.childNodes[3].childNodes[3];

const taskDescription = parentNode.childNodes[3].childNodes[5];

const taskType = parentNode.childNodes[3].childNodes[7].childNodes[1];

const submitButton = parentNode.childNodes[5].childNodes[1];

const updatedData = {

taskTitle: taskTitle.innerHTML,

taskDescription: taskDescription.innerHTML,

taskType: taskType.innerHTML,

};

let stateCopy = state.taskList;

stateCopy = stateCopy.map((task) =>

task.id === targetID

? {

id: task.id,

title: updatedData.taskTitle,

description: updatedData.taskDescription,

type: updatedData.taskType,

url: task.url,

}

: task

);

state.taskList = stateCopy;

updateLocalStorage();

taskTitle.setAttribute("contenteditable", "false");

taskDescription.setAttribute("contenteditable", "false");

taskType.setAttribute("contenteditable", "false");

submitButton.setAttribute("onclick", "openTask.apply(this, arguments)");

submitButton.setAttribute("data-bs-toggle", "modal");

submitButton.setAttribute("data-bs-target", "#showTask");

submitButton.innerHTML = "Open Task";

};

const searchTask = (e) => {

if (!e) e = window.event;

while (taskContents.firstChild) {

taskContents.removeChild(taskContents.firstChild);

}

const resultData = state.taskList.filter(({ title }) =>

title.toLowerCase().includes(e.target.value.toLowerCase())

);

resultData.map((cardData) =>

taskContents.insertAdjacentHTML("beforeend", htmlTaskContent(cardData))

);

};