**Project 1: My Digital Portfolio**

**What I Did:**

I created a responsive portfolio website to showcase my skills, projects, and achievements. The website is designed to be user-friendly and works seamlessly on different devices, from desktops to mobile phones.

**Key Features:**

* **Sections:** The website includes dedicated sections for "About Me," "Projects," "Skills," and "Contact."
* **Responsive Design:** I used CSS Flexbox and Grid to ensure the layout adapts to various screen sizes.
* **Navigation:** A simple navigation bar at the top allows users to jump to any section with a single click.
* **Optional:** I added a photo gallery to visually highlight my work and a testimonials section to showcase feedback from others.

**Code Highlights:**

**HTML:**

<nav>

<ul>

<li><a href="#about">About Me</a></li>

<li><a href="#projects">Projects</a></li>

<li><a href="#skills">Skills</a></li>

<li><a href="#contact">Contact</a></li>

</ul>

</nav>

<section id="about">

<h2>About Me</h2>

<p>Hi, I'm [Your Name], a passionate web developer...</p>

</section>

**CSS:**

.projects-grid {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));

gap: 20px;

}

**How It Works:**

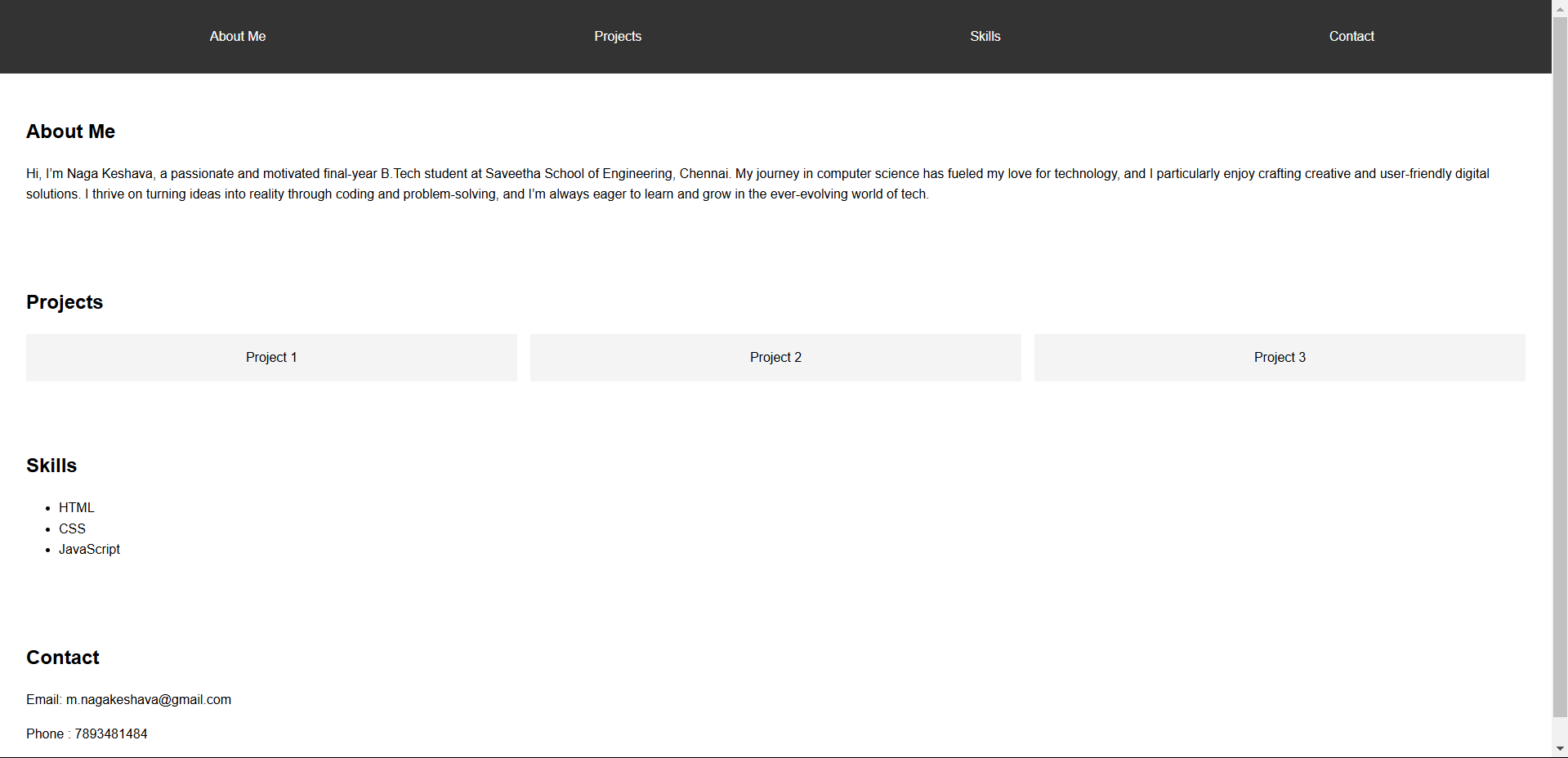
* The "About Me" section introduces who I am and what I do.
* The "Projects" section displays my work with brief descriptions and links.
* The "Skills" section lists my technical abilities, like HTML, CSS, and JavaScript.
* The "Contact" section provides a way for visitors to reach out to me.

**Tools and Technologies Used:**

* HTML5 for structuring the content.
* CSS3 (Flexbox and Grid) for styling and responsiveness.
* JavaScript for adding interactivity (like smooth scrolling).
* VS Code as my primary code editor.
* Google Fonts and Font Awesome for typography and icons.

**Screenshots:**

Here are some screenshots of the final result:



**Project 3: Task Master**

**What I Did:**

I built a dynamic to-do list application that helps users organize their tasks efficiently. The app allows users to add, mark as complete, and delete tasks, with all data saved in the browser’s local storage.

**Key Features:**

* **Task Management:** Users can add new tasks, mark them as complete, or delete them.
* **Local Storage:** Tasks are saved in the browser, so they persist even after the page is refreshed.
* **Styling:** I used CSS to make the app visually appealing and easy to use.
* **Optional:** I added a drag-and-drop feature to let users reorder tasks.

**Code Highlights:**

**JAVASCRIPT:**

function addTask() {

const taskInput = document.getElementById("task-input");

const taskList = document.getElementById("task-list");

if (taskInput.value !== "") {

const li = document.createElement("li");

li.textContent = taskInput.value;

taskList.appendChild(li);

taskInput.value = "";

saveTasks();

}

}

function saveTasks() {

const tasks = [];

document.querySelectorAll("#task-list li").forEach(task => {

tasks.push(task.textContent);

});

localStorage.setItem("tasks", JSON.stringify(tasks));

}

**How It Works:**

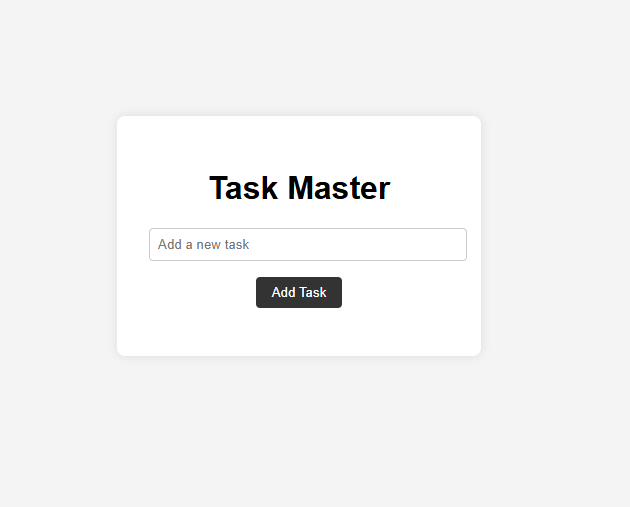
* Users type a task into the input field and click "Add" to include it in the list.
* Completed tasks can be checked off, and unwanted tasks can be deleted.
* All tasks are stored in the browser’s local storage, so they’re still there when the user returns.

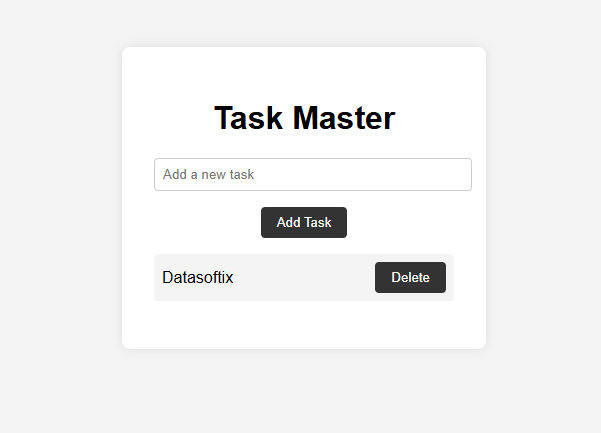
**Tools and Technologies Used:**

* HTML5 for the app’s structure.
* CSS3 for styling and layout.
* JavaScript for handling task management and local storage.
* VS Code for writing and testing the code.

**Screenshots:**

Here are some screenshots of the final result:





**Project 6: The Art Explorer**

**What I Did:**

I designed a responsive image gallery that displays artwork in a clean and organized way. The gallery includes hover effects and a modal window to view images in larger detail. Users can also filter images by categories.

**Key Features:**

* **Grid Layout:** Images are displayed in a responsive grid that adjusts to different screen sizes.
* **Hover Effects:** When users hover over an image, it slightly enlarges or shows a caption.
* **Modal Window:** Clicking on an image opens it in a larger view within a modal window.
* **Filters:** Users can filter images by categories like nature, abstract, or portraits.
* **Optional:** I added a feature to allow users to upload new images to the gallery.

**Code Highlights:**

**HTML:**

<div class="gallery-grid">

<img src="image1.jpg" alt="Art 1" class="gallery-item">

<img src="image2.jpg" alt="Art 2" class="gallery-item">

</div>

<div id="modal" class="modal">

<span class="close">&times;</span>

<img class="modal-content" id="modal-img">

</div>

**JAVASCRIPT:**

document.querySelectorAll(".gallery-item").forEach(item => {

item.addEventListener("click", () => {

const modal = document.getElementById("modal");

const modalImg = document.getElementById("modal-img");

modal.style.display = "block";

modalImg.src = item.src;

});

});

**How It Works:**

* The gallery loads images in a grid format.
* Users can click on any image to open it in a modal for a closer look.
* The filter buttons allow users to view only specific categories of images.

**Tools and Technologies Used:**

* HTML5 for the gallery structure.
* CSS3 for styling and hover effects.
* JavaScript for the modal window and filtering functionality.
* VS Code for development.

**Screenshots:**

Here are some screenshots of the gallery:

