Name : Halkhoree Tanushree

StudentID : 21613049

Link to Project : https://portfolio-fusion.netlify.app/

AI Integration Task & Conversation Logs – Portfolio Fusion

1. Reflection – Portfolio Fusion

I set out to develop a polished and expressive portfolio for my Portfolio Fusion project that showcased my personal style and web development capabilities. I envisioned the website to be elegant, fluid, and truly "me." For me to do this, I created a glam-themed homepage incorporating dynamic flip-card project exhibits, floating sparkles, descending petals, and a typing introduction.

One of the standout features of this project was the use of AI as a technical and creative collaborator. I invested a lot of time using ChatGPT to deal with layout and responsiveness issues, generate ideas, and create reusable animations. As an example, in my Rock Paper Scissors app, I utilized AI to assist in creating typing visuals, sparkling and bloom effects, and JavaScript-powered project interactivity like flip cards and delaying game responses.

AI was a guide, not a bypass, though. Every snippet I got had been reviewed and edited by me, with layouts modified to fit my theme and logic improved to meet my objectives. The dark-themed toggle is a fantastic example of this; while AI supported me in constructing the structure, I was accountable for rewriting the design as well as making sure it was responsive and understandable.

By generating more comprehensive prompts and expanding on feedback, this project enabled me to learn how to engage with AI with greater efficiency. Furthermore, it increased my grasp of media queries, JavaScript DOM events, and CSS transitions. I learned that although AI can speed up the development process, programmers are still charged with innovation and attention to detail.

In the end, I constructed a professional, fun, and personal portfolio that encompasses both dark as well as light topics. I'm satisfied with how this project progressed and how AI transformed into a genuine participant during the design process.

2 .AI Interaction Log – Portfolio Fusion

Prompt 1:

Initial Prompt: My homepage is so dull. How do i make it sparkly?

AI Response Summary:

AI suggested two styles: falling sparkles and twinkling static sparkles. Provided JavaScript and CSS to create randomly placed sparkles using setInterval().

My adjustment:

I suggested that the sparkles remain in place and continue to sparkle. The animation trace was removed, and I used @keyframes twinkle with opacity and scale.

Final Result :

Now, sparkles slowly shimmer and float elegantly in the background, not drawing focus away from the main text.

Prompt 2:

Initial Prompt:

I want a typing animation for the heading, how to do that?

AI Response Summary:

AI provided a CSS solution using @keyframes typing and steps(), along with overflow: hidden and a blinking border.

My Adjustment:

I applied a class="typing" to <h1>with responsive font sizes and modified the motion speed.

Final Result:

"Hello, I'm Tanushree" is now automatically typed out on the website.

Prompt 3:

Initial Prompt:

How to make the 4 cards interactive?

AI Response Summary:

AI suggested using transform: rotateY(180deg) and 3D perspective.

My Adjustment:

I used.card-inner to wrap each project card, separated the content between.card-front and.card-back, and applied the pink theme and shadowing to both sides.

Final Result:

Each of the cards has tools and project connections on the back that are shown when it is hovered over.

Prompt 4 :

Initial Prompt:

I want to add an animation on load

AI Response Summary:
AI gave JavaScript and CSS to create <div> elements containing 🌸 and animate them down the screen.

My Adjustment:

I used setInterval() for launching petals over time after inserting different font sizes and left space to make it look authentic.

Final Result :

Every time the website loads, flowers fall beautifully in the background, creating a dreamy, glam touch.

Prompt 5:

Initial Prompt:

How to make RPS result like a real game?

AI Response Summary:
AI recommended delaying the display of the result using setTimeout() and clearing out the choices temporarily.

My Adjustment:

I used tension text, updated the player/computer screen lines, and wrapped the logic behind a 3-second wait.

Final Result:

RPS Game now shows “Processing your move…” before revealing results, improving user engagement.

3. Before/After Code Snippets

Dark Mode Toggle

AI Suggestion (Before) :

```
document.querySelector('button').addEventListener('click', () => {
  document.body.classList.toggle('dark');
});
```

My final Version (After) :

```
const toggleButton = document.createElement('button');
toggleButton.textContent = '🌙';
Object.assign(toggleButton.style, {
  position: 'fixed',
  top: '10px',
  right: '10px',
  padding: '0.5rem 1rem',
  backgroundColor: '#a0134d',
  color: 'white',
  borderRadius: '20px',
});
document.body.appendChild(toggleButton);
toggleButton.addEventListener('click', () => {
  document.body.classList.toggle('dark');
  toggleButton.textContent = document.body.classList.contains('dark') ? '☀️' : '🌙';
});
```

Reason for changing : I wanted a floating button with an unique design, smooth navigation, and icon toggle.

Flip Card Animation

AI Suggestion (Before) :

```
.card:hover {
  transform: rotateY(180deg);
}
```

My Final Version (After):

HTML

```
<div class="project-card">
  <div class="card-inner">
    <div class="card-front">...</div>
    <div class="card-back">...</div>
  </div>
</div>
```

CSS

```
.project-card {
  perspective: 1000px;
}
.card-inner {
  transform-style: preserve-3d;
  transition: transform 0.6s;
}
.project-card:hover .card-inner {
  transform: rotateY(180deg);
}
```

Reason for changing :

I incorporated inner wrappers for responsiveness, shadows, and front/back control despite the AI suggested flipping the entire card.

Typing Intro Animation

AI Suggestion (Before):

```
@keyframes typing {
  from { width: 0; }
  to { width: 100%; }
}
```

My Final Version (After):

```
.typing {
  white-space: nowrap;
  overflow: hidden;
  border-right: 3px solid #a0134d;
  animation: typing 4s steps(40, end), blink 0.8s step-end infinite;
}
@keyframes blink {
  50% { border-color: transparent; }
}
```

Reason for Change:

I wrapped the blinking cursor and typing action in a class="typing" so that it could be utilized in <h1>
.

Flower and Sparkle Animation

AI Suggestion (Before):

```
// One petal fall example
let petal = document.createElement('div');
petal.innerText = '🌸';
```

My Final Version (After):

```
function createPetal() {
  const petal = document.createElement('div');
  petal.className = 'petal';
  petal.textContent = '🌸';
  petal.style.left = Math.random() * 100 + 'vw';
  petal.style.fontSize = (Math.random() * 10 + 10) + 'px';
  petal.style.animationDuration = (Math.random() * 5 + 5) + 's';
  document.body.appendChild(petal);
  setTimeout(() => petal.remove(), 10000);
}
```

Reason for change : For a more sophisticated, natural animation, I used generated size, length, spacing, and cleanup.

Evidence of Prompt Evolution

Prompt Evolution : Sparkles

Initial Prompt:

“My homepage is so dull. How do I make it sparkly?”

Follow-up Prompt:

“I want sparkles that twinkle and stay in place, not fall.”

Final Prompt:

“Give me JavaScript and CSS to place ✨ sparkles randomly on the screen and animate them using scale and opacity with @keyframes twinkle.”

Prompt Evolution: Typing Intro

Initial Prompt:

“I want a typing animation for the heading. How to do that?”

Follow-up Prompt:

“Can you make it type across with a blinking cursor?”

Final Prompt:

“Create a CSS typing animation using steps() and blinking border for my <h1> and style it responsively.”

Prompt Evolution: Flip Cards

Initial Prompt:

“How to make the 4 project cards interactive?”

Follow-up Prompt:

“I want the cards to flip on hover and show tools or links on the back.”

Final Prompt:

“Build flip cards with .card-inner, .card-front, and .card-back using rotateY, flexbox layout, and glam styling.”

Prompt Evolution: Petal Fall on Load

Initial Prompt:

“I want to add an animation on load.”

Follow-up Prompt:

“Make petals fall with different sizes and natural flow.”

Final Prompt:

“Use setInterval() and randomised CSS to create 🌸 petals that fall from the top with varying sizes and duration.”

Prompt Evolution: RPS Result Timing

Initial Prompt:

“How to make the RPS result like a real game?”

Follow-up Prompt:

“Add suspense — show ‘Processing…’ before result appears.”

Final Prompt:

“Wrap my result logic in a 3-second setTimeout() and update the DOM with suspense text and choices before showing outcome.”