Phase 4: Development Part 2

Web development technologies such as HTML, CSS, and JavaScript are used in combination to create a platform that displays real-time water fountain status. Here is how each technology for smart water fountain plays a role:

(1) HTML (Hyper Text Markup Language):

- HTML Structure: Start by creating the basic HTML structure for your platform. Define elements for displaying water flow rate and malfunction alerts.
- Structure: HTML provides the basic structure of the web page. It defines the different elements on the page such as headings, paragraphs, and most importantly, the containers where the real-time data will be displayed.
- Content: HTML elements are used to represent the content of the web page, including text, images, and other media.
- In the context of displaying water fountain status, HTML is used to define the layout of the status platform. It sets up the overall structure and content of the page, including placeholders for real-time data like flow rate and malfunction alerts.

Coding:

```
<!DOCTYPE html>

<html>

<head>

  <title>Water Fountain Status</title>

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

</head>

<body>

  <div class="status-container">

    <h1>Water Fountain Status</h1>

    <div class="flow-rate">

      <h2>Flow Rate: <span id="flow-rate-value">0</span> GPM</h2>

    </div>
```

```
      <div class="malfunction-alert">
        <h2>Malfunction Alert: <span id="malfunction-status">No</span></h2>
      </div>
    </div>
    <script src="script.js"></gscript>
</body>
</html>
```

(2) <u>CSS (Cascading Style Sheets):</u>

- CSS Styling: Style your HTML elements using CSS to make the platform visually appealing and user-friendly.
- Style: CSS is responsible for the visual presentation of the HTML elements. It defines the layout, colors, fonts, and other visual aspects of the web page.
- Responsive Design: CSS can be used to make the platform responsive, ensuring that it looks good and functions well on various devices and screen sizes.
- For the water fountain status platform, CSS styles the HTML elements to create an aesthetically pleasing and user-friendly interface. It ensures that the platform is visually appealing and easy to navigate.

<u>Coding:</u>

```
body {
    font-family: Arial, sans-serif;
    background-color: #f4f4f4;
}
.status-container {
    max-width: 600px;
    margin: 0 auto;
    padding: 20px;
    background-color: #ffffff;
```

```
    border-radius: 10px;
    box-shadow: 0px 0px 10px 0px rgba(0,0,0,0.1);
}
.flow-rate, .malfunction-alert {
    margin-top: 20px;
}
```

(3) JavaScript:

- Use JavaScript to fetch and display real-time data on the platform.
- Interactivity: JavaScript adds interactivity to the web page. It can handle real-time data, respond to user actions, and update the content dynamically without requiring a page refresh.
- Asynchronous Requests: JavaScript can make asynchronous requests to APIs or servers, allowing the platform to fetch real-time data without reloading the entire page.
- Dynamic Updates: JavaScript can dynamically update the content of HTML elements, enabling the display of real-time data such as water flow rate and malfunction alerts.
- In the context of the water fountain status platform, JavaScript is used to fetch real-time data from sensors or APIs. It updates the HTML elements dynamically, providing a real-time view of the water fountain status. JavaScript can also handle user interactions and perform actions based on user input.
- By combining these technologies, developers can create a dynamic and responsive platform that displays real-time water fountain status to users, providing an engaging and informative user experience.

Coding:

```
function updateStatus() {
    // Simulated real-time data (replace this with actual API calls)
    const flowRate = (Math.random() * 10).toFixed(2); // Random flow rate between 0 and 10 GPM
    const isMalfunction = Math.random() > 0.9; // 10% chance of malfunction
    // Update HTML elements with real-time data
```

```
    document.getElementById('flow-rate').textContent = flowRate;
    document.getElementById('malfunction').textContent = isMalfunction ? 'Yes' : 'No';
}
// Update data every 2 seconds (2000 milliseconds)
setInterval(updateStatus, 2000);
```

In this setup, the 'updateStatus' function simulates fetching real-time data. You should replace the simulated data with actual data fetched from your water fountain sensors or API endpoints. The JavaScript code updates the flow rate and malfunction status every 2 seconds, providing a real-time experience for the users.

Remember to replace the simulated data with actual data sources for a functional and accurate water fountain status platform.