Cross-Site Scripting (XSS)

To be protected against DoS attacks, I made changes in the httpd.js. HTML Entity Encoding and HTML Attribute Encoding.

encodeHtmlEntities

We convert the following characters into a safe form to be aware of the XSS attacks: (& to &, Convert < to <, Convert > to >, Convert " to ", Convert ' to ').

This protects us from code injection through the url

```
// Function to encode HTML entities
function encodeHtmlEntities(unsafe) {
  return unsafe
    .replace(/&/g, "&amp;")
    .replace(/</g, "&lt;")
    .replace(/>/g, "&gt;")
    .replace(/"/g, "&quot;")
    .replace(/'/g, "&#x27;");
}
```

encodeHtmlAttributes

We take a string as input and encode all non-alphanumeric characters in the &#xHH; format. Alphanumeric characters (letters A-Z, a-z, and digits 0-9) will remain unencoded.

```
function encodeHtmlAttributes(unsafe) {
  return unsafe
    .split('')
    .map(char => {
      const charCode = char.charCodeAt(0);
      if (
        (charCode >= 48 && charCode <= 57) || // Digits 0-9
        (charCode >= 65 && charCode <= 90) || // Uppercase letters A-Z
        (charCode >= 97 && charCode <= 122)   // Lowercase letters a-z
      ) {
        // Characters A-Z, a-z, 0-9 remain unencoded
        return char;
      } else {
        // Encode other characters using the HTML Entity format &#xHH;
        return `&#x${charCode.toString(16).toUpperCase()};`;
      }
    })
    .join('');
}
```