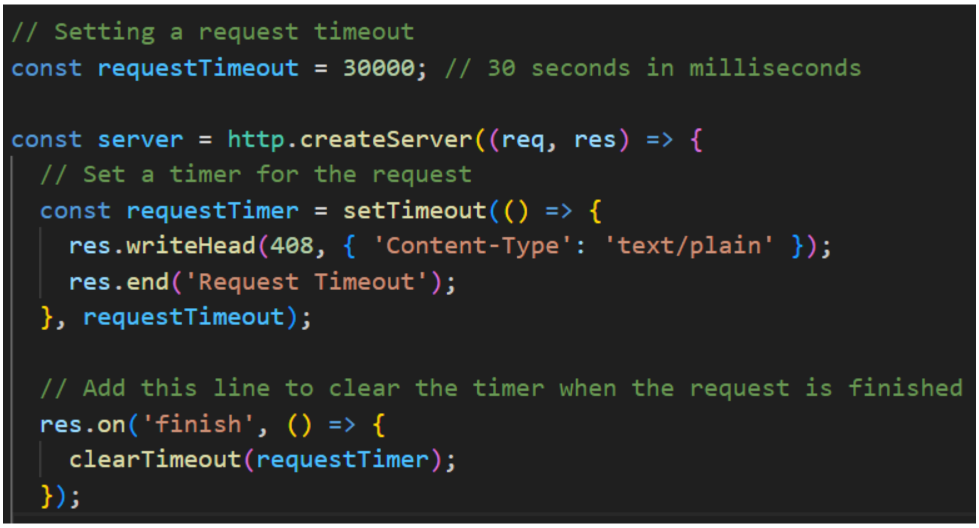
Report Denial of Service (DoS)

To be protected against DoS attacks, I made changes in the httpd.js. A timeout defense and a client-side validation defense.

Timeout

The change I made was to protect the web server from being overwhelmed by long or maliciously crafted requests. Additionally, it helps against resource management attacks and Slowloris attacks.

I did set a timeout for incoming requests to prevent them from taking too long. If a request takes too long (more than 30 seconds), it will be terminated.

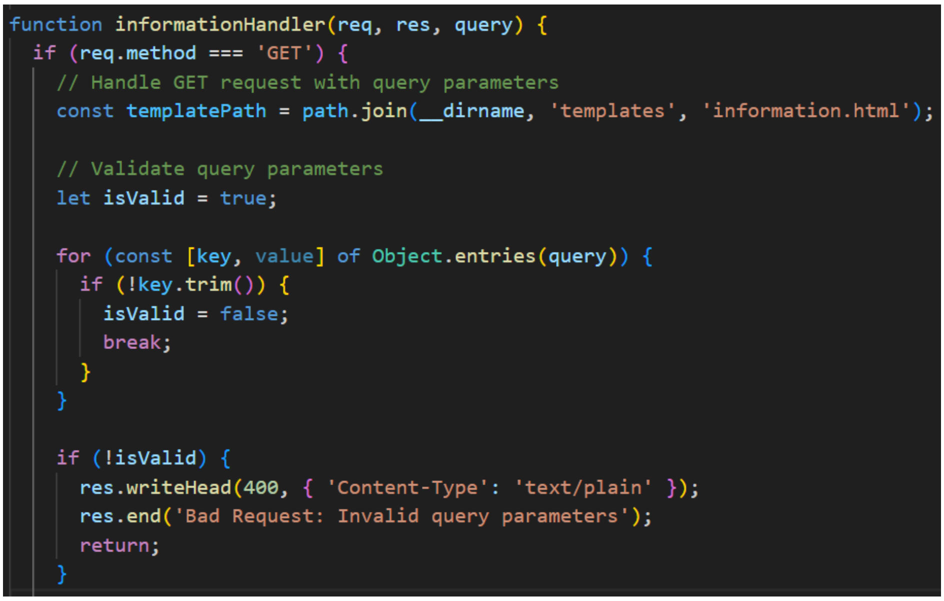


```
// Setting a request timeout
const requestTimeout = 30000; // 30 seconds in milliseconds

const server = http.createServer((req, res) => {
  // Set a timer for the request
  const requestTimer = setTimeout(() => {
    res.writeHead(408, { 'Content-Type': 'text/plain' });
    res.end('Request Timeout');
  }, requestTimeout);

  // Add this line to clear the timer when the request is finished
  res.on('finish', () => {
    clearTimeout(requestTimer);
  });
```

Client-side validation

The change I made was adding a simple validation in a for loop that checks if all URL parameters are non-empty strings. If any parameter is empty, it responds with an HTTP 400 (Bad Request) status code and an error message.

With this implementation, requests containing empty URL parameters will be rejected.



```
function informationHandler(req, res, query) {
  if (req.method === 'GET') {
    // Handle GET request with query parameters
    const templatePath = path.join(__dirname, 'templates', 'information.html');

    // Validate query parameters
    let isValid = true;

    for (const [key, value] of Object.entries(query)) {
      if (!key.trim()) {
        isValid = false;
        break;
      }
    }

    if (!isValid) {
      res.writeHead(400, { 'Content-Type': 'text/plain' });
      res.end('Bad Request: Invalid query parameters');
      return;
    }
```