The language used in the provided code is JavaScript, specifically using the Express.js framework for handling HTTP requests and responses.

There are a couple of potential security issues with this code:

1. **XML External Entity (XXE) Attack**:

   - The most significant issue here is the option `{ noent: true }` used with `libxml.parseXml`. This option allows the parser to process XML entities. This can lead to an XML External Entity (XXE) attack, where attackers can use XML input to read local files, make requests to internal resources, or cause a denial of service.

2. **Lack of Input Validation**:

   - Before parsing the XML or passing it to any function (like `addToFavorites`), you should validate the input. Without validation, there's potential for other types of attacks, such as sending malicious XML payloads that could exploit vulnerabilities in the processing logic or cause excessive resource consumption (also known as "Billion Laughs" attacks when related to XML).

3. **Potential for Denial of Service (DoS)**:

   - Parsing XML can be resource-intensive. An attacker might send large XML documents to exhaust server resources, leading to a DoS condition.

4. **Global Variable Usage**:

   - The variable `favorite` is implicitly global because it's not declared with `let`, `const`, or `var`. This can lead to unexpected behaviors and vulnerabilities if another part of the code modifies or accesses this global variable.

To enhance security:

- Disable XML entity processing by setting `{ noent: false }` or not including the `noent` option at all.

- Validate XML input before processing it. Ensure that it adheres to the expected structure and doesn't contain unexpected data or elements.

- Limit the size of incoming XML payloads to prevent potential DoS attacks.

- Declare variables with `let`, `const`, or `var` to ensure they are not global and to maintain proper scope.

- Always consult the documentation and security guidelines specific to the libraries and frameworks you're using.

Remember that XXE attacks can have severe implications, so it's crucial to be diligent when handling and parsing XML data.