

⇒ **O problema: conexões de baixa latência de cliente-servidor e servidor-cliente:**

A web tem sido construída com base no conhecido paradigma de solicitação/resposta de HTTP. Um cliente carrega uma página da web e, em seguida, nada acontece até que o usuário clique na próxima página. Por volta de 2005, o AJAX começou a deixar a web mais dinâmica. Mesmo assim, toda a comunicação HTTP era direcionada pelo cliente, o que exigia interação do usuário ou sondagem periódica para carregar novos dados do servidor.

As tecnologias que permitem que o servidor envie dados ao cliente no mesmo momento em que constata que novos dados estão disponíveis foram usadas por algum tempo. Elas eram conhecidas por nomes como "Push" ou "Comet". Um dos problemas mais comuns para criar a ilusão de uma conexão iniciada pelo servidor é a chamada sondagem longa. Com a sondagem longa, o cliente abre uma conexão HTTP com o servidor que permanece aberta até que a resposta seja enviada. Sempre que tem novos dados, o servidor envia a resposta (outras técnicas envolvem Flash, solicitações XHR multipart e os chamados htmlfiles). A sondagem longa e as outras técnicas funcionam muito bem. Você as utiliza todos os dias em aplicativos como o chat do Gmail.

No entanto, todas essas soluções compartilham um problema: elas carregam a sobrecarga de HTTP, que não é adequada para aplicativos de baixa latência. Pense em jogos com vários jogadores no navegador ou em qualquer outro jogo on-line com um componente em tempo real.

FONTE: <https://www.html5rocks.com/pt/tutorials/websockets/basics/>

⇒ **Apresentando WebSocket: trazendo soquetes para a web:**

A especificação WebSocket define uma API que estabelece conexões de "soquete" entre um navegador da web e um servidor. Em outras palavras, há uma conexão persistente entre o cliente e o servidor e ambas as partes podem começar a enviar dados a qualquer momento.

(...)

Use o WebSocket sempre que precisar de uma conexão quase em tempo real de baixa latência entre o cliente e o servidor. Tenha em mente que isso pode envolver a reformulação do modo como você cria os aplicativos de servidor com um novo foco em tecnologias como filas de eventos. Alguns exemplos de casos de uso:

- Jogos on-line de vários jogadores
- Aplicativos de chat
- Links para esportes ao vivo
- Atualização em tempo real de redes sociais

(...)

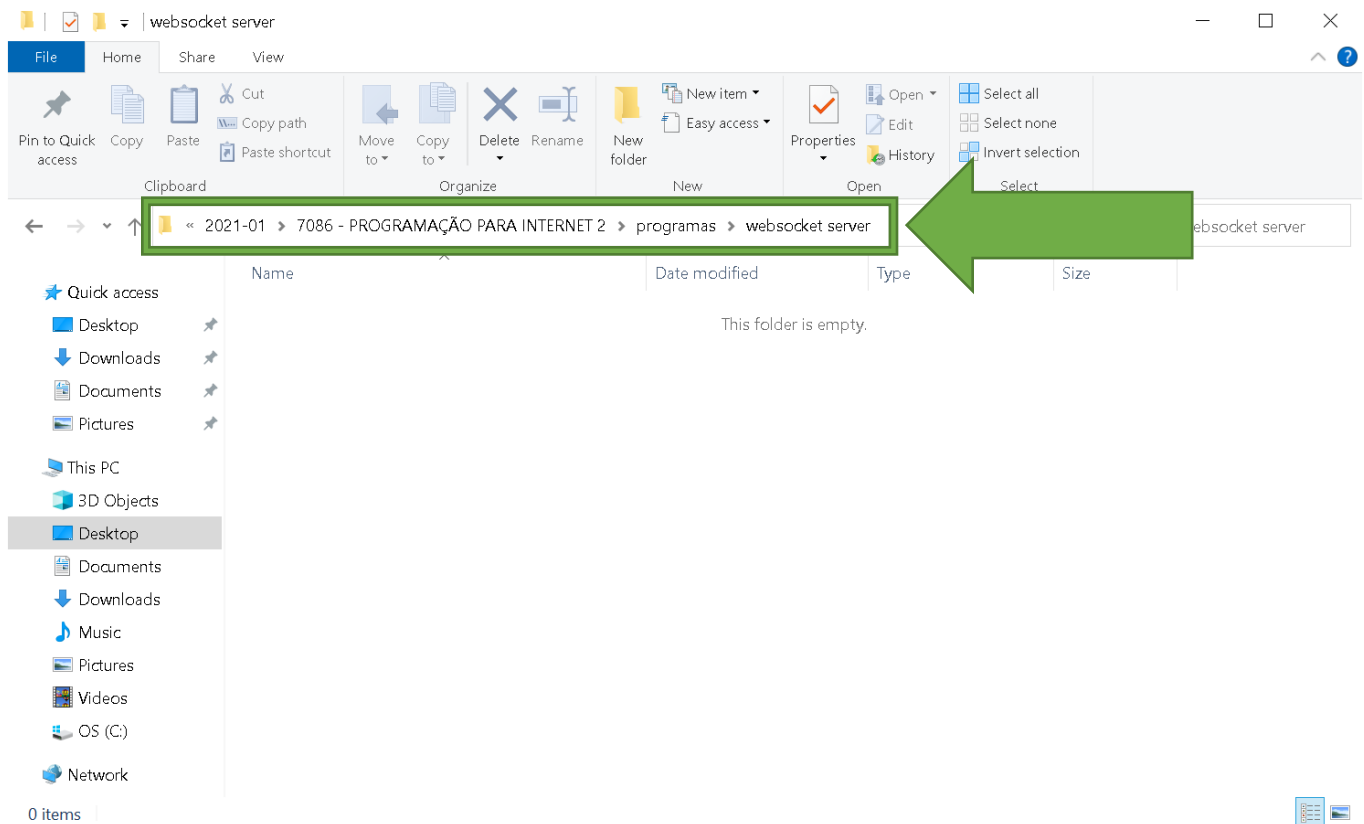
Sendo um protocolo moderno, a comunicação de origem cruzada está integrada diretamente no WebSocket. Embora você ainda deva se comunicar somente com clientes e servidores confiáveis, o WebSocket permite a comunicação entre partes de qualquer domínio. O servidor decide se disponibilizará seu serviço para todos os clientes ou somente para os que residem em um conjunto de domínios bem definidos.

(...)

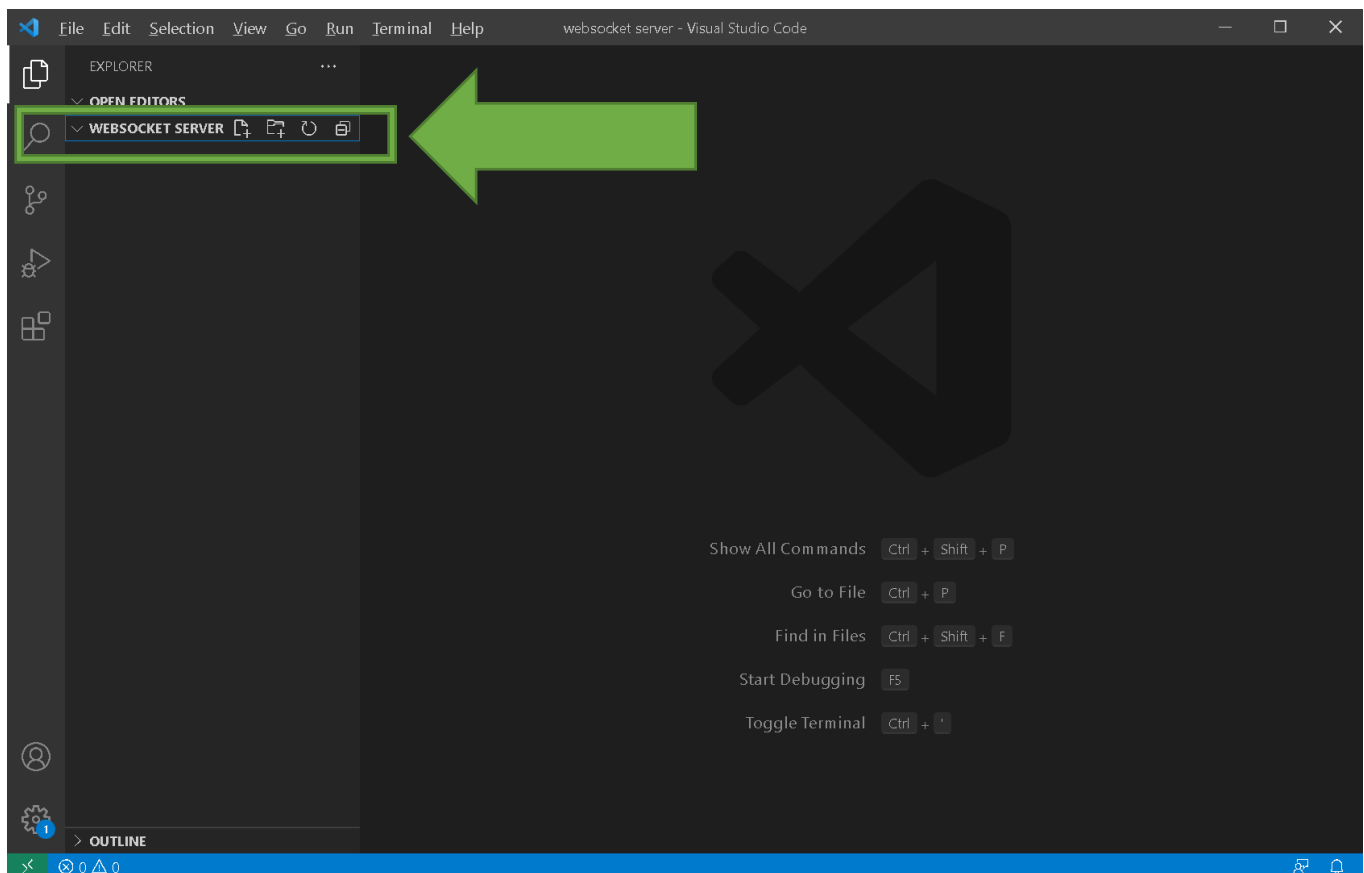
O WebSocket ainda é uma tecnologia jovem e não foi completamente implementada em todos os navegadores. No entanto, você pode usar o WebSocket hoje com as bibliotecas que usam um dos [fallbacks](#) mencionados acima sempre que o WebSocket não estiver disponível. Uma biblioteca que se tornou muito popular nesse domínio é a [socket.io](#), que é fornecida com um cliente e uma implementação de servidor do protocolo e inclui fallbacks (o socket.io não oferece suporte para mensagens binárias ainda, segundo dados de fevereiro de 2012). Há também soluções comerciais como [PusherApp](#) que podem ser facilmente integradas em qualquer ambiente da web fornecendo uma API HTTP para enviar mensagens WebSocket aos clientes. Devido à solicitação HTTP extra, sempre haverá sobrecarga adicional em comparação com o WebSocket puro.

ATIVIDADE

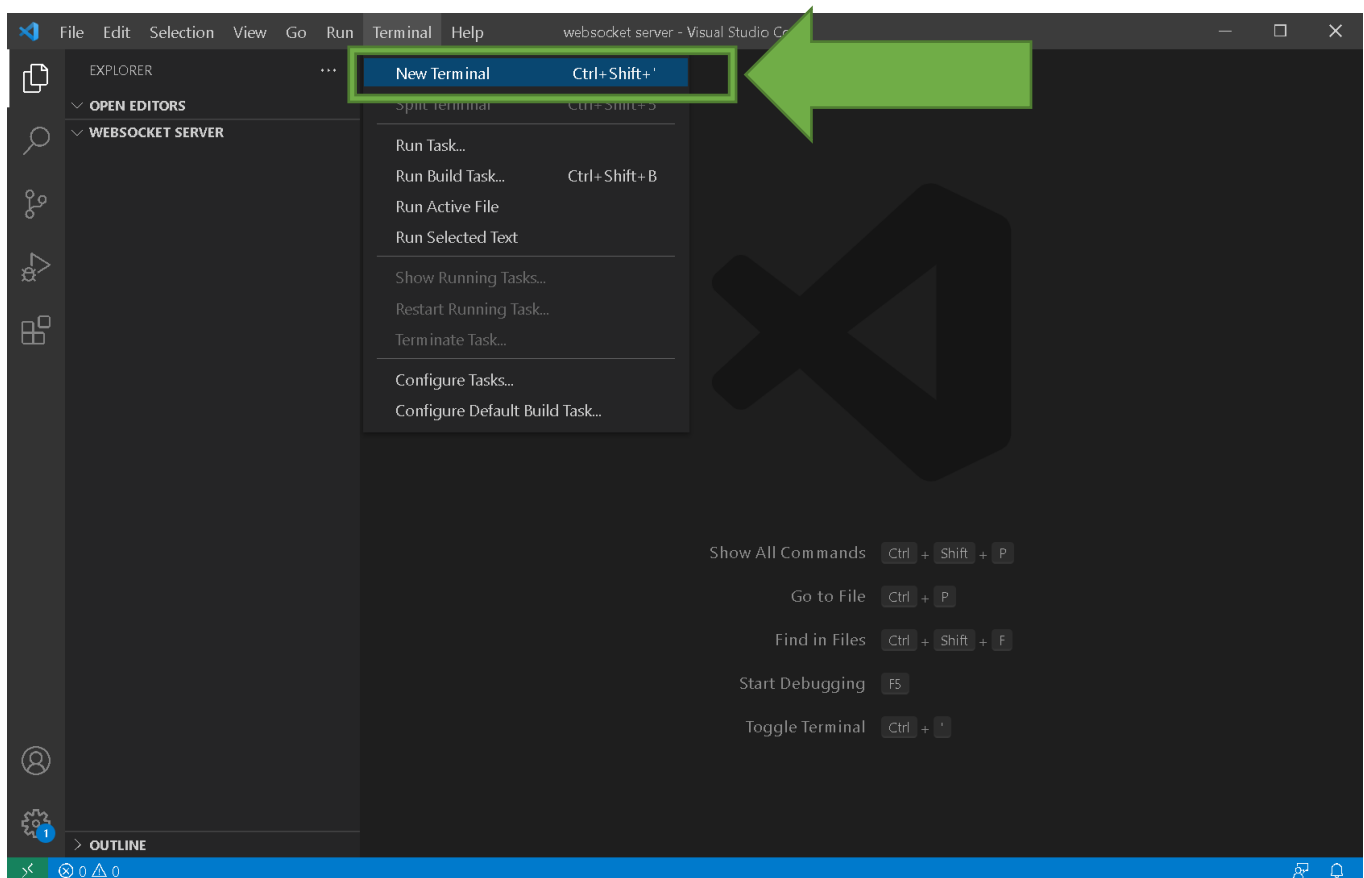
⇒ Crie a pasta “websocket server” onde irá criar o projeto:



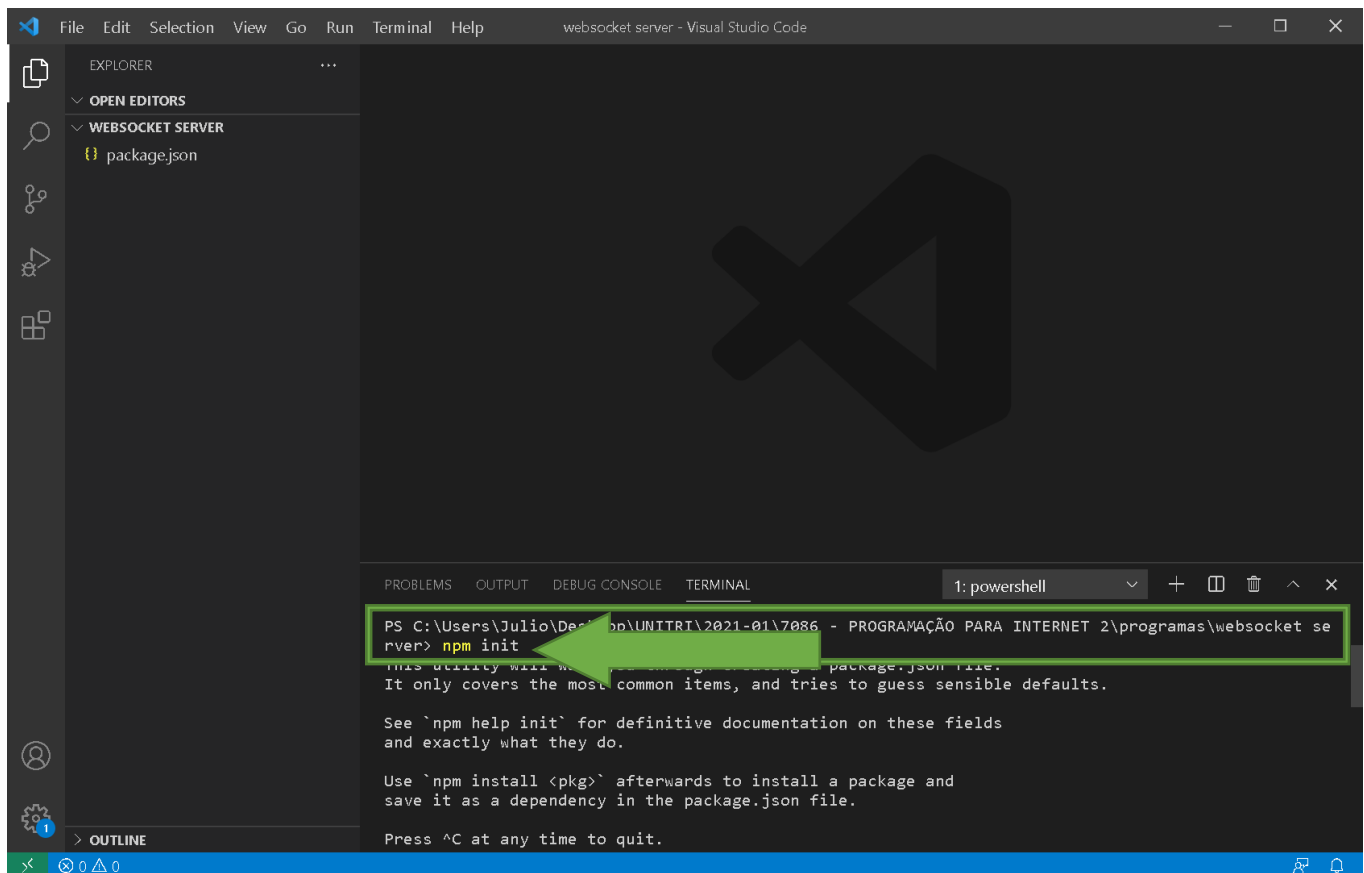
⇒ Abra a pasta no VS Code:



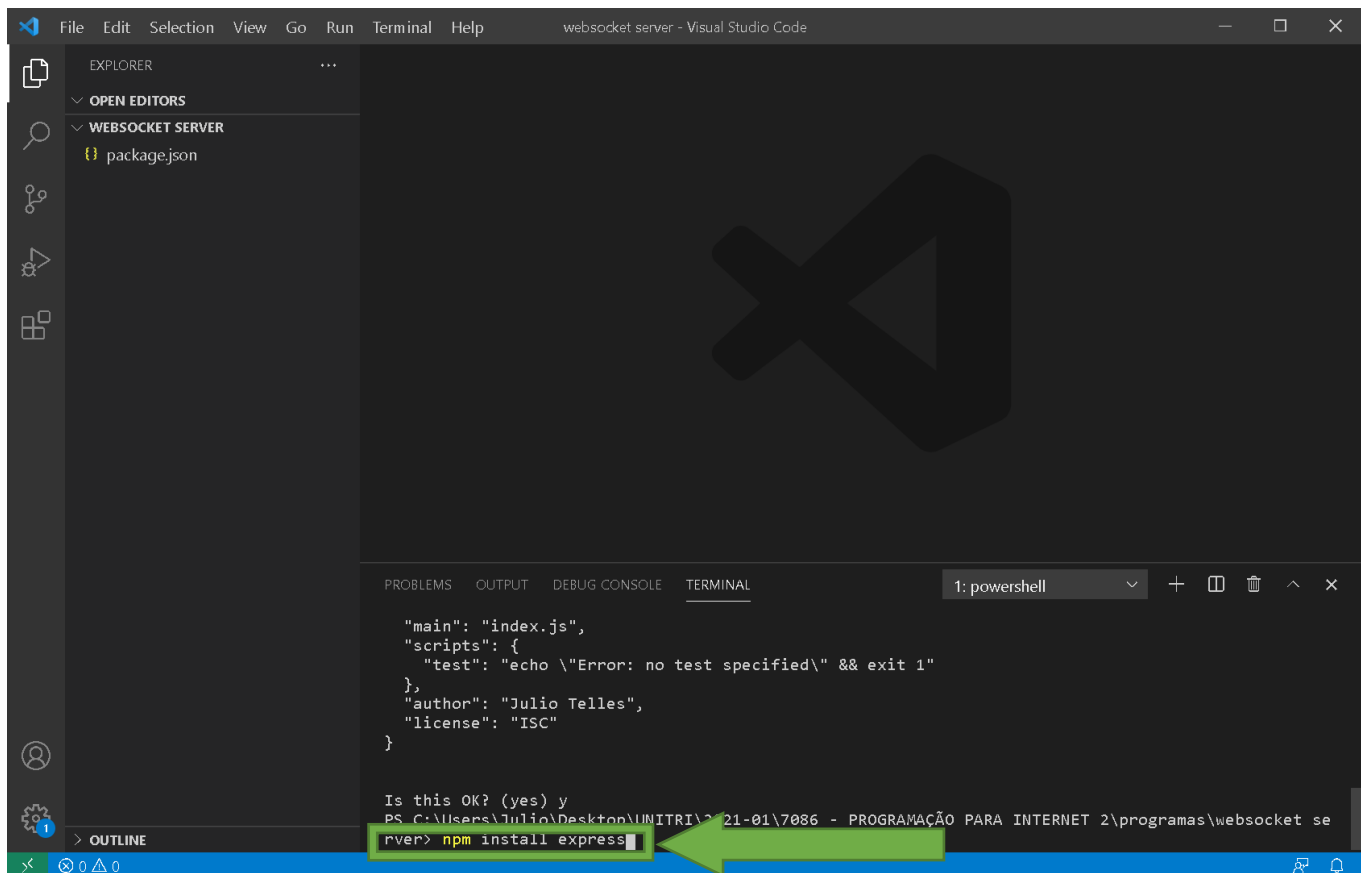
⇒ Abra o terminal:



⇒ Crie o JSON para o servidor:



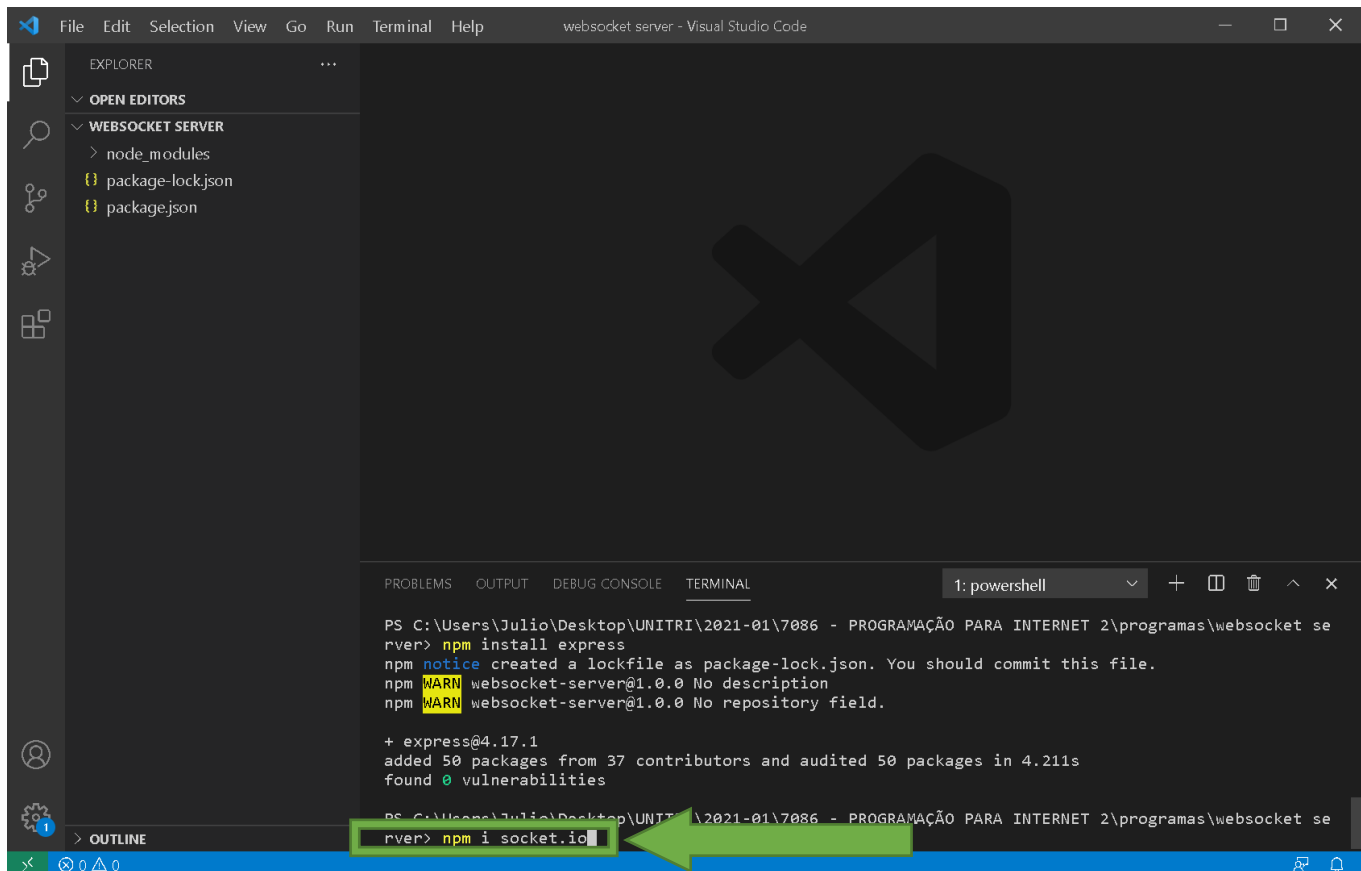
⇒ Instale os pacotes:



Visual Studio Code interface showing the initial state of a project. The Explorer sidebar shows the file structure with 'package.json' under 'WEBSOCKET SERVER'. The Terminal panel is open, displaying the contents of 'package.json' and a prompt to run 'npm install express'.

```
"main": "index.js",
"scripts": {
  "test": "echo \"Error: no test specified\" && exit 1"
},
"author": "Julio Telles",
"license": "ISC"
}

Is this OK? (yes) y
PS C:\Users\Julio\Desktop\UNITRI\2021-01\7086 - PROGRAMAÇÃO PARA INTERNET 2\programas\websocket se
rver> npm install express
```



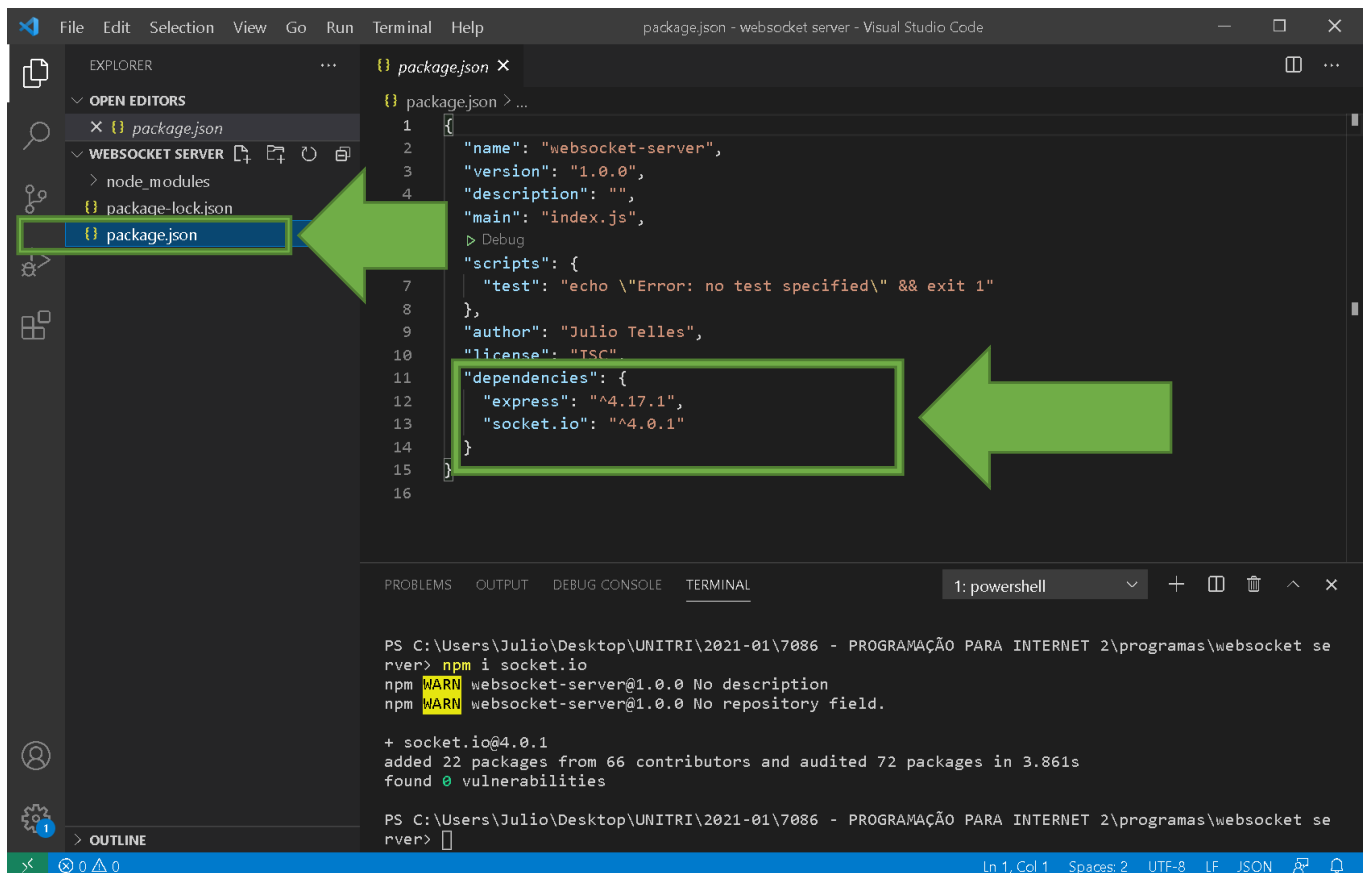
Visual Studio Code interface showing the result of running 'npm install express'. The Explorer sidebar now shows 'node_modules', 'package-lock.json', and 'package.json'. The Terminal panel shows the output of the command, including warnings and the installation of 'express@4.17.1'.

```
PS C:\Users\Julio\Desktop\UNITRI\2021-01\7086 - PROGRAMAÇÃO PARA INTERNET 2\programas\websocket se
rver> npm install express
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN websocket-server@1.0.0 No description
npm WARN websocket-server@1.0.0 No repository field.

+ express@4.17.1
added 50 packages from 37 contributors and audited 50 packages in 4.211s
found 0 vulnerabilities

PS C:\Users\Julio\Desktop\UNITRI\2021-01\7086 - PROGRAMAÇÃO PARA INTERNET 2\programas\websocket se
rver> npm i socket.io
```

⇒ Prossiga com os ajustes abaixo:

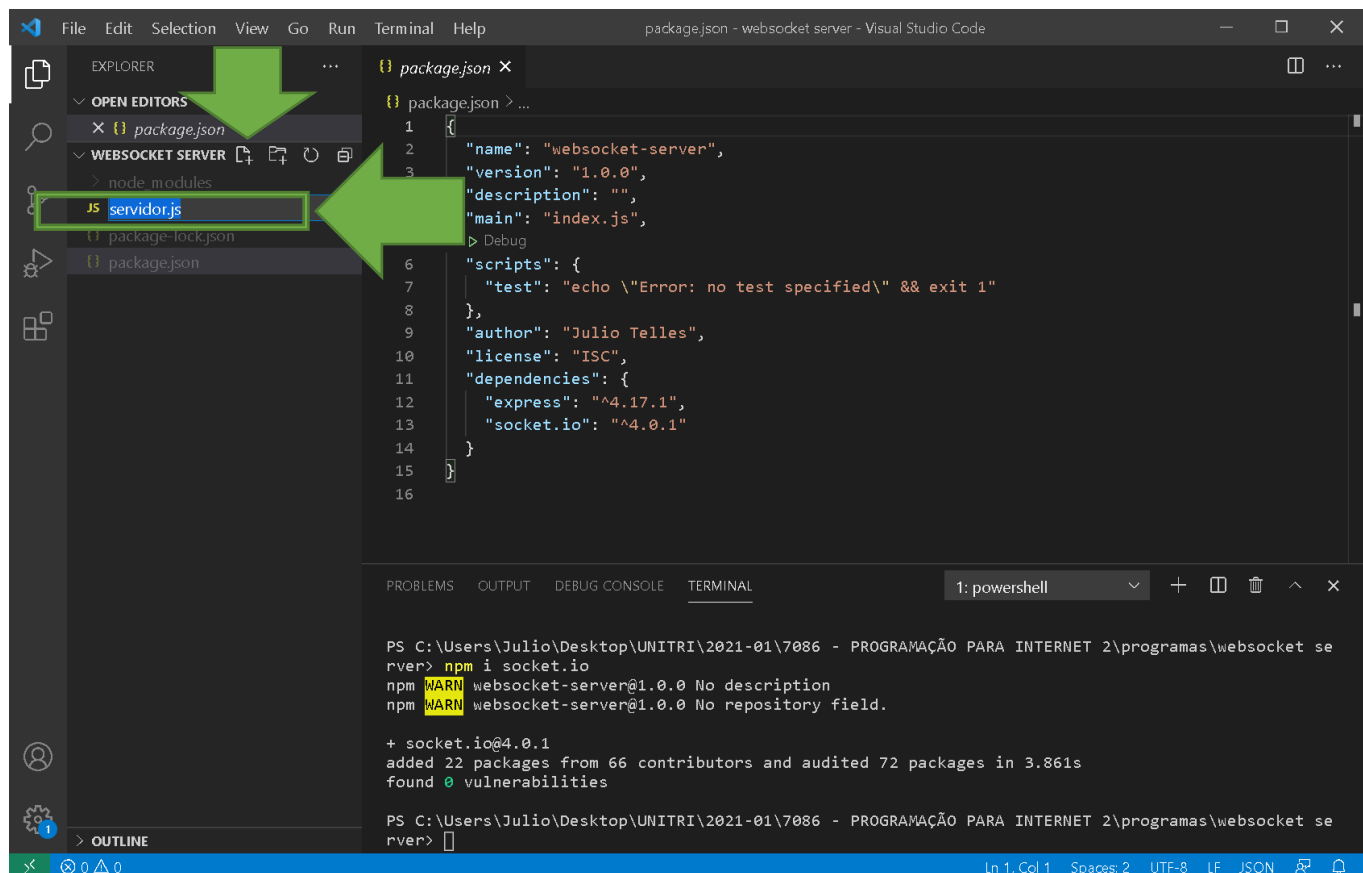


This screenshot shows the Visual Studio Code interface with the `package.json` file open in the editor. The Explorer sidebar on the left shows the project structure, with `package.json` highlighted. The editor displays the following JSON configuration:

```
1 {  
2   "name": "websocket-server",  
3   "version": "1.0.0",  
4   "description": "",  
5   "main": "index.js",  
6   "scripts": {  
7     "test": "echo \"Error: no test specified\" && exit 1"  
8   },  
9   "author": "Julio Telles",  
10  "license": "ISC",  
11  "dependencies": {  
12    "express": "^4.17.1",  
13    "socket.io": "^4.0.1"  
14  }  
15 }  
16
```

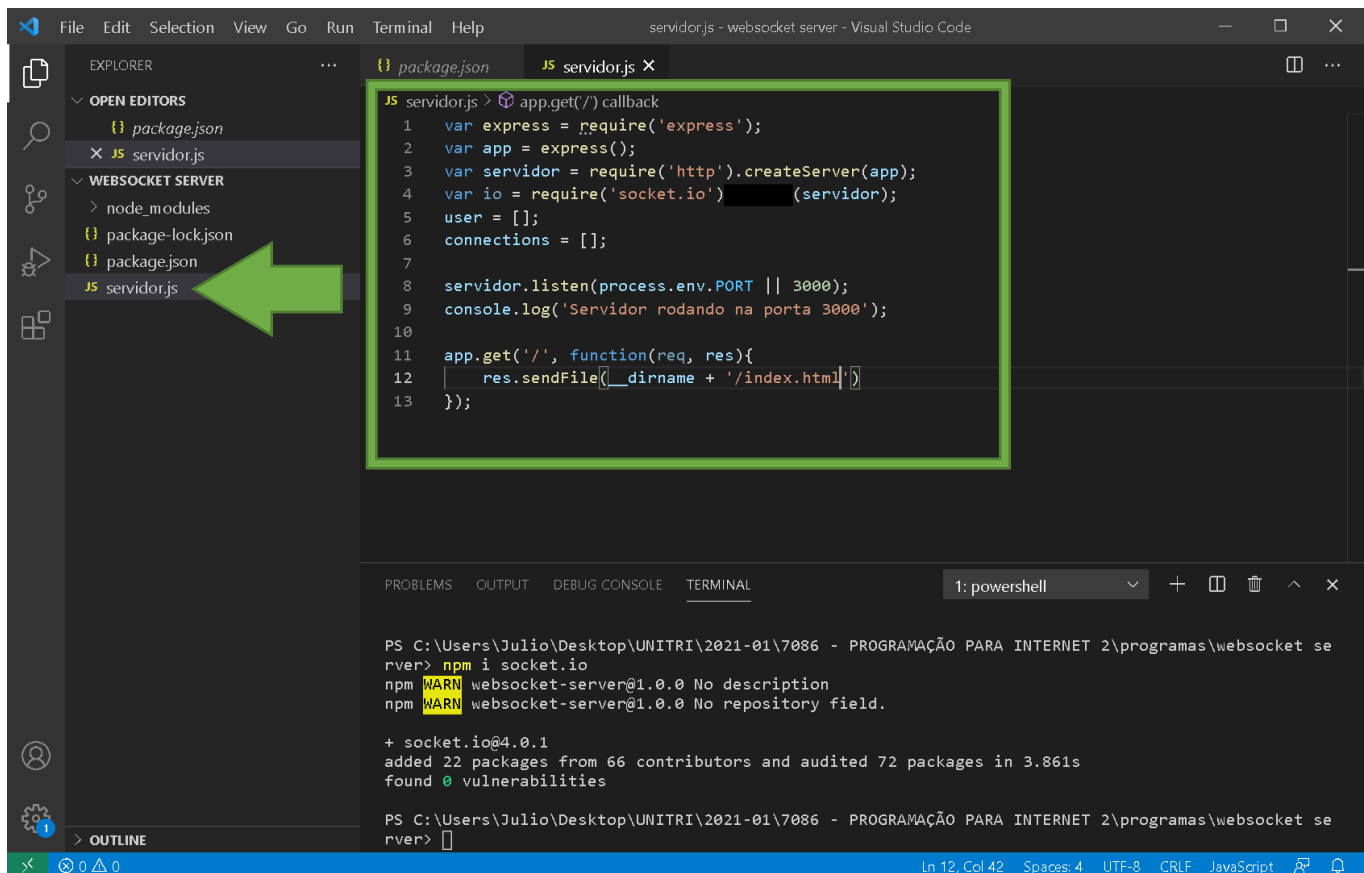
Two green arrows point to the `package.json` file in the Explorer and the `dependencies` section in the editor. The terminal at the bottom shows the output of the command `npm i socket.io`:

```
PS C:\Users\Julio\Desktop\UNITRI\2021-01\7086 - PROGRAMAÇÃO PARA INTERNET 2\programas\websocket se  
rver> npm i socket.io  
npm WARN websocket-server@1.0.0 No description  
npm WARN websocket-server@1.0.0 No repository field.  
  
+ socket.io@4.0.1  
added 22 packages from 66 contributors and audited 72 packages in 3.861s  
found 0 vulnerabilities  
  
PS C:\Users\Julio\Desktop\UNITRI\2021-01\7086 - PROGRAMAÇÃO PARA INTERNET 2\programas\websocket se  
rver>
```



This screenshot shows the Visual Studio Code interface with the `package.json` file open in the editor. The Explorer sidebar on the left shows the project structure, with `servidor.js` highlighted. The editor displays the same `package.json` configuration as the previous screenshot. Two green arrows point to the `servidor.js` file in the Explorer and the `package.json` file in the editor. The terminal at the bottom shows the same output as the previous screenshot.

⇒ Ajuste o arquivo “servidor.js”:



```
JS servidor.js > app.get('/') callback
1  var express = require('express');
2  var app = express();
3  var servidor = require('http').createServer(app);
4  var io = require('socket.io')(servidor);
5  user = [];
6  connections = [];
7
8  servidor.listen(process.env.PORT || 3000);
9  console.log('Servidor rodando na porta 3000');
10
11 app.get('/', function(req, res){
12   res.sendFile(__dirname + '/index.html');
13 });
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: powershell

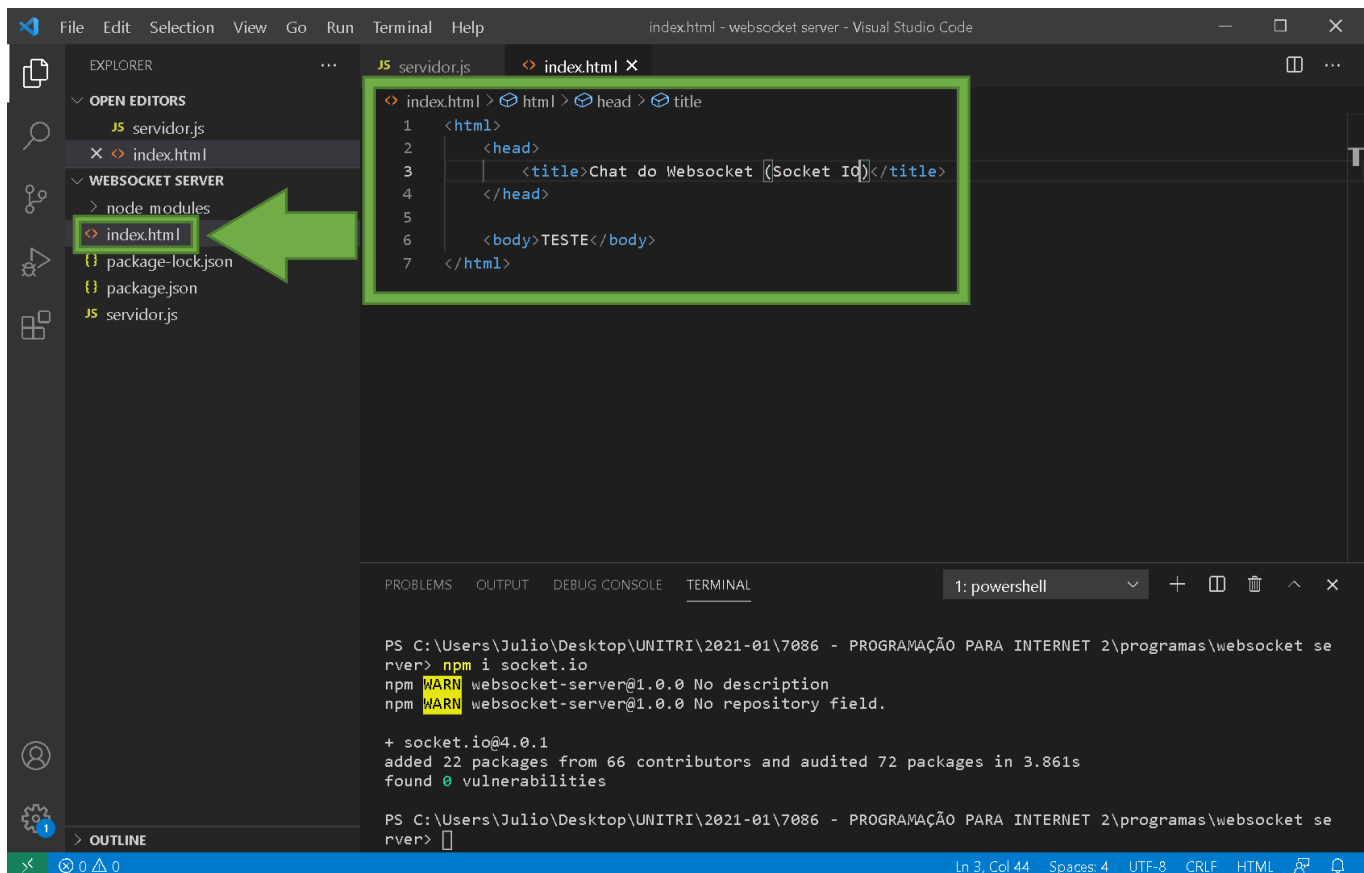
```
PS C:\Users\Julio\Desktop\UNITRI\2021-01\7086 - PROGRAMAÇÃO PARA INTERNET 2\programas\websocket se
rver> npm i socket.io
npm WARN websocket-server@1.0.0 No description
npm WARN websocket-server@1.0.0 No repository field.

+ socket.io@4.0.1
added 22 packages from 66 contributors and audited 72 packages in 3.861s
found 0 vulnerabilities

PS C:\Users\Julio\Desktop\UNITRI\2021-01\7086 - PROGRAMAÇÃO PARA INTERNET 2\programas\websocket se
rver>
```

Ln 12, Col 42 Spaces: 4 UTF-8 CRLF JavaScript

⇒ Crie e ajuste o arquivo “index.html”:



```
JS servidor.js index.html X
1  <html>
2    <head>
3      <title>Chat do Websocket [Socket Id]</title>
4    </head>
5
6    <body>TESTE</body>
7  </html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: powershell

```
PS C:\Users\Julio\Desktop\UNITRI\2021-01\7086 - PROGRAMAÇÃO PARA INTERNET 2\programas\websocket se
rver> npm i socket.io
npm WARN websocket-server@1.0.0 No description
npm WARN websocket-server@1.0.0 No repository field.

+ socket.io@4.0.1
added 22 packages from 66 contributors and audited 72 packages in 3.861s
found 0 vulnerabilities

PS C:\Users\Julio\Desktop\UNITRI\2021-01\7086 - PROGRAMAÇÃO PARA INTERNET 2\programas\websocket se
rver>
```

Ln 3, Col 44 Spaces: 4 UTF-8 CRLF HTML

⇒ Teste o funcionamento do servidor:

The image shows a development environment in Visual Studio Code and a web browser. In VS Code, the Explorer sidebar on the left shows a project structure with files: `servidor.js`, `index.html`, `node_modules`, `package-lock.json`, and `package.json`. The `index.html` file is open in the editor, displaying the following HTML code:

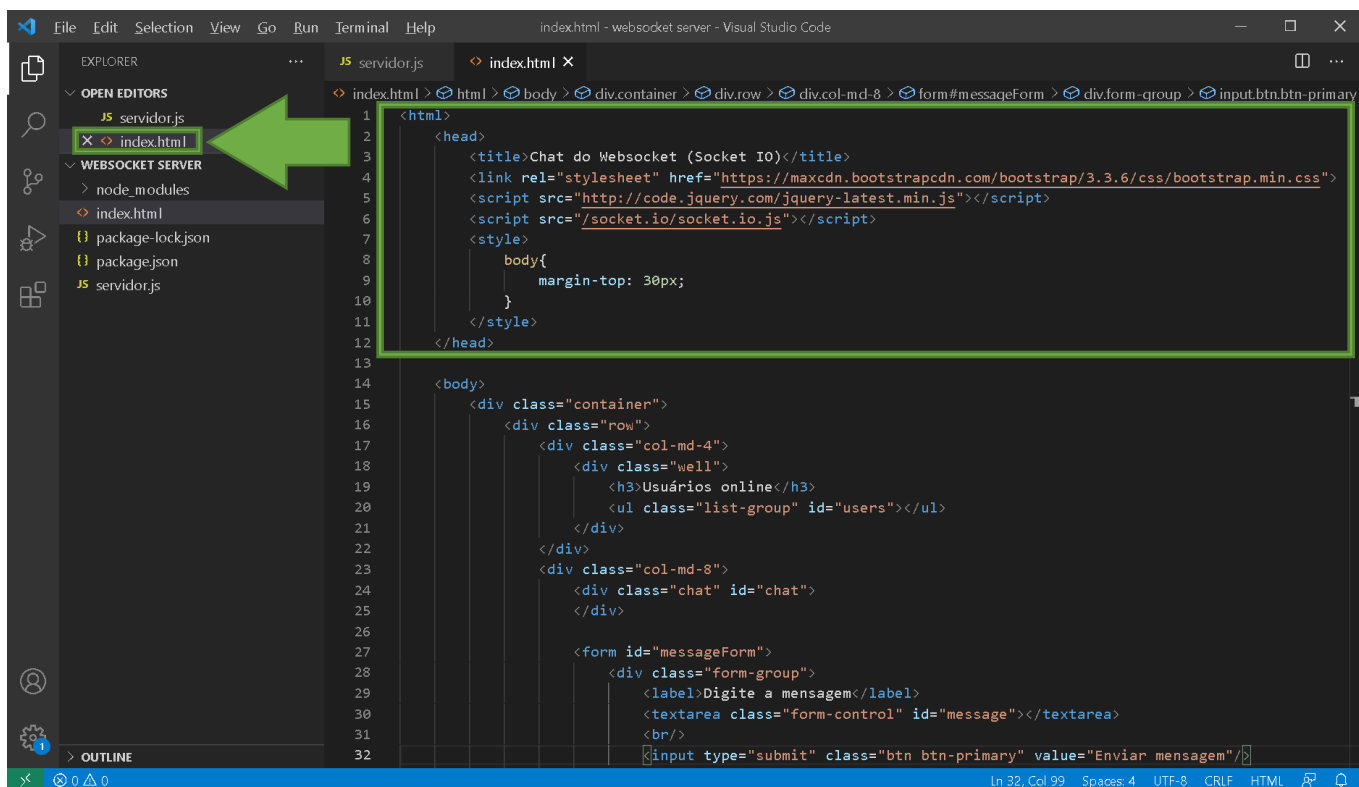
```
1 <html>
2   <head>
3     <title>Chat do Websocket (Socket IO)</title>
4   </head>
5
6   <body>TESTE</body>
7 </html>
```

The Terminal at the bottom shows the command `node servidor` being executed, with the output:

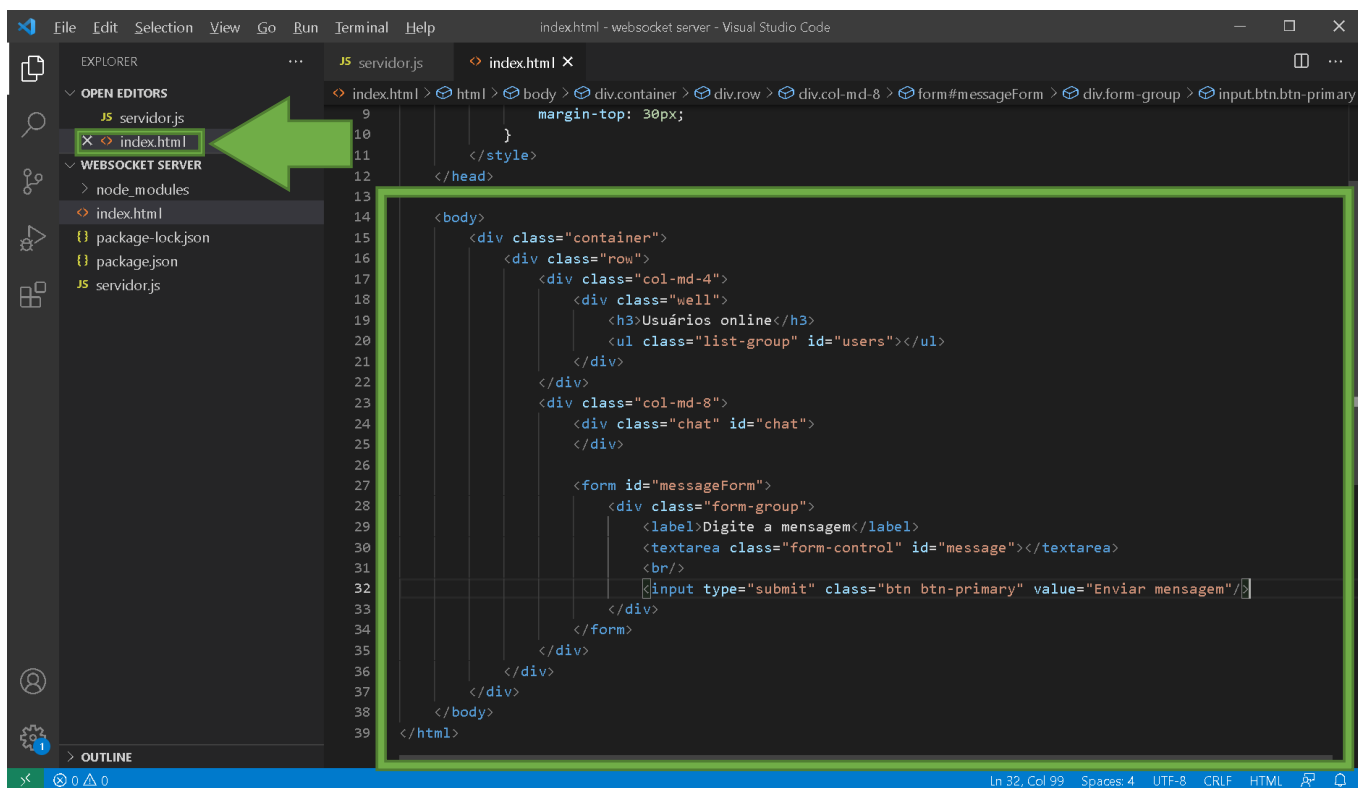
```
at Object.<anonymous> (C:\Users\Julio\Desktop\UNITRI\2021-01\7086 - PROGRAMAÇÃO PARA INTERNET 2\programas\websocket server\servidor.js:4:31)
at Module._compile (internal/modules/cjs/loader.js:1063:30)
at Object.Module._extensions..js (internal/modules/cjs/loader.js:1092:10)
at Module.load (internal/modules/cjs/loader.js:928:32)
at Function.Module._load (internal/modules/cjs/loader.js:769:14)
at Function.executeUserEntryPoint [as runMain] (internal/modules/run_main.js:72:12)
at internal/main/run_main_module.js:17:47
PS C:\Users\Julio\Desktop\UNITRI\2021-01\7086 - PROGRAMAÇÃO PARA INTERNET 2\programas\websocket server> node servidor
Servidor rodando na porta 3000
```

Two green arrows point from the terminal output to the browser. The browser window, titled "Chat do Websocket (Socket IO)", shows the address bar with `localhost:3000` highlighted by a green box. The page content displays the text "TESTE".

⇒ Realize os ajustes no arquivo “index.html”:

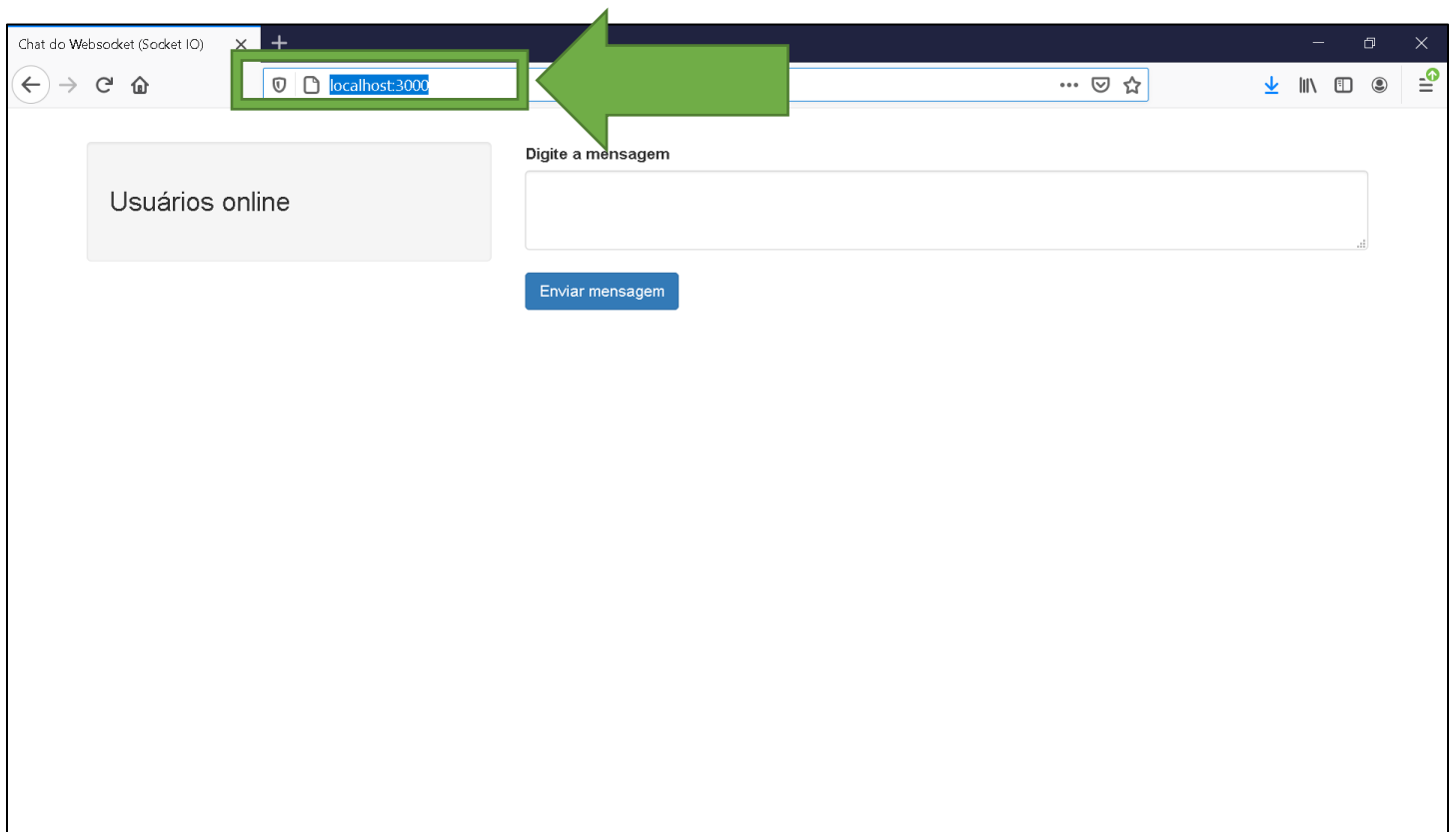


```
1 <html>
2   <head>
3     <title>Chat do Websocket (Socket IO)</title>
4     <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css">
5     <script src="http://code.jquery.com/jquery-latest.min.js"></script>
6     <script src="/socket.io/socket.io.js"></script>
7   </head>
8   <body>
9     <div class="container">
10      <div class="row">
11        <div class="col-md-4">
12          <div class="well">
13            <h3>Usuários online</h3>
14            <ul class="list-group" id="users"></ul>
15          </div>
16        </div>
17        <div class="col-md-8">
18          <div class="chat" id="chat">
19          </div>
20          <form id="messageForm">
21            <div class="form-group">
22              <label>Digite a mensagem</label>
23              <textarea class="form-control" id="message"></textarea>
24              <br/>
25              <input type="submit" class="btn btn-primary" value="Enviar mensagem"/>
26            </div>
27          </form>
28        </div>
29      </div>
30    </div>
31  </body>
32 </html>
```

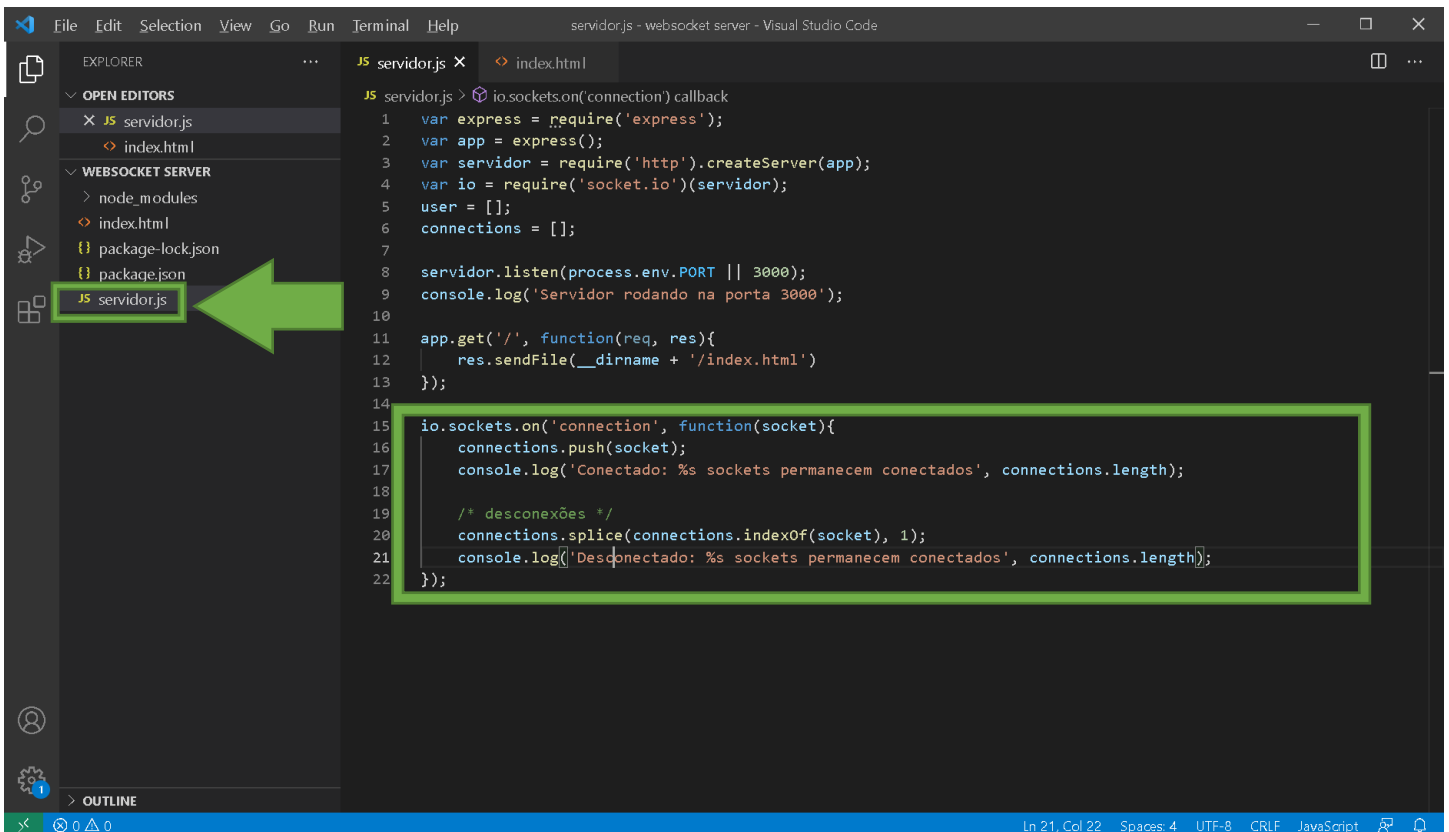


```
9   <div class="container">
10     <div class="row">
11       <div class="col-md-4">
12         <div class="well">
13           <h3>Usuários online</h3>
14           <ul class="list-group" id="users"></ul>
15         </div>
16       </div>
17       <div class="col-md-8">
18         <div class="chat" id="chat">
19         </div>
20         <form id="messageForm">
21           <div class="form-group">
22             <label>Digite a mensagem</label>
23             <textarea class="form-control" id="message"></textarea>
24             <br/>
25             <input type="submit" class="btn btn-primary" value="Enviar mensagem"/>
26           </div>
27         </form>
28       </div>
29     </div>
30   </div>
31 </body>
32 </html>
```

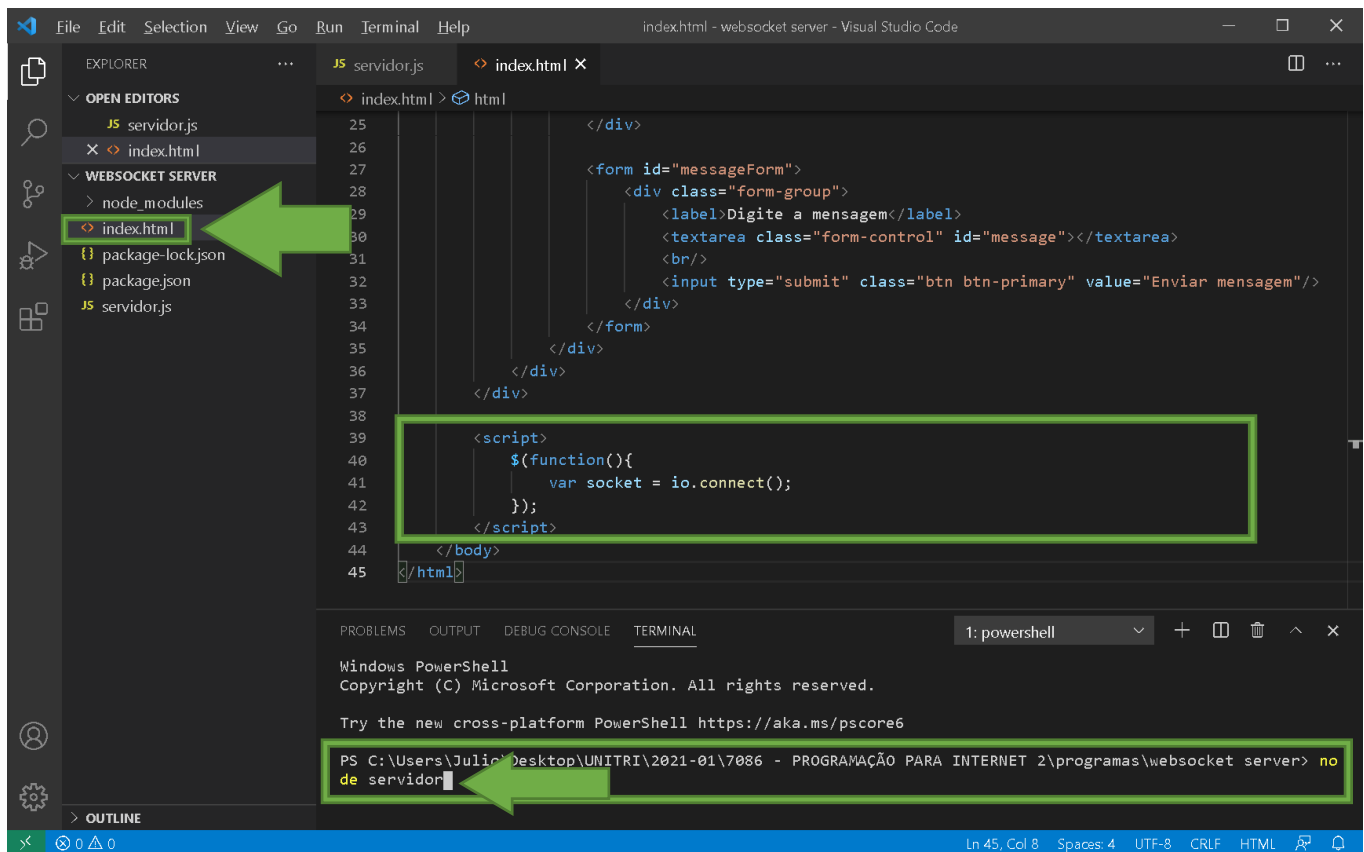

⇒ Atualize o navegador:



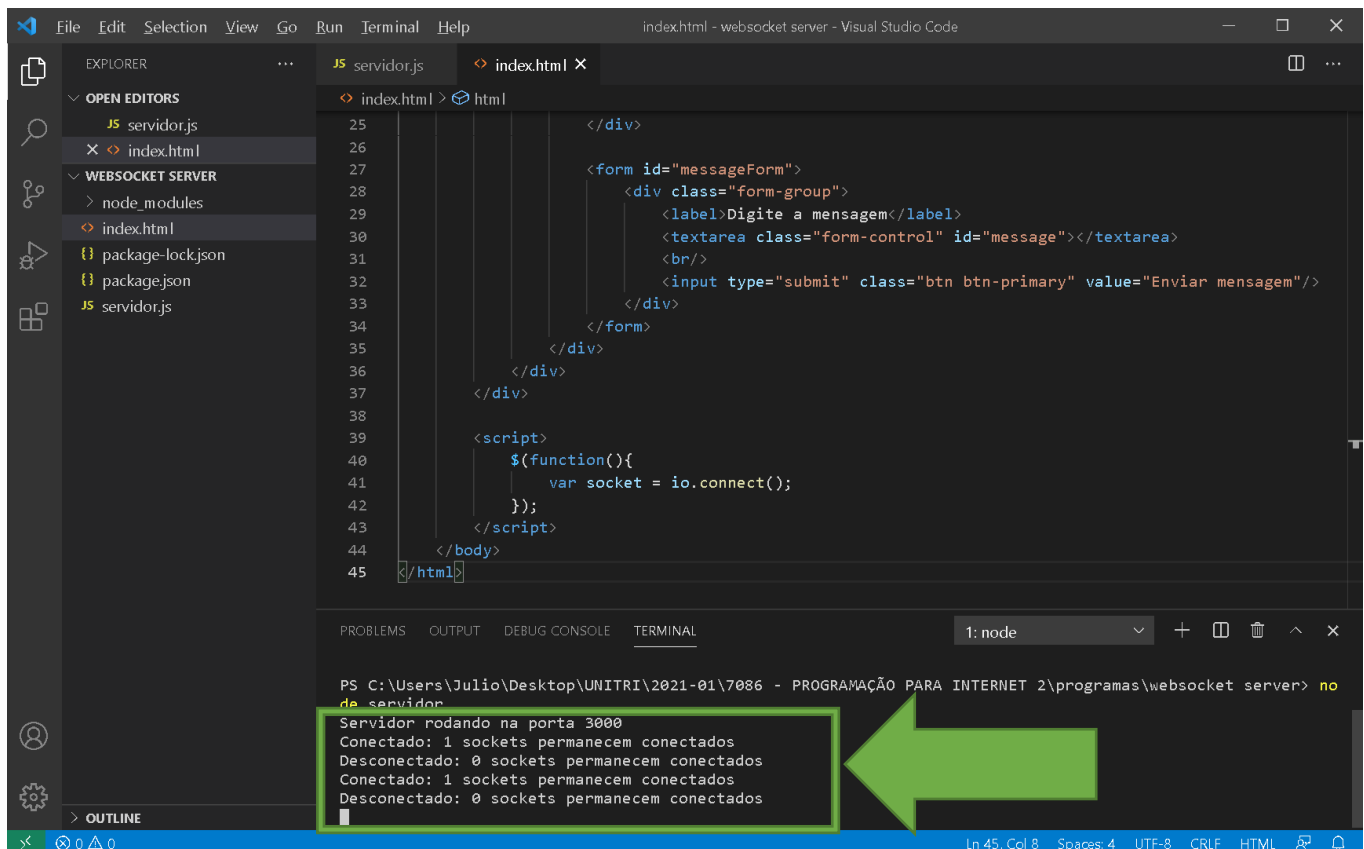
⇒ Ajuste o arquivo “servidor.js”:



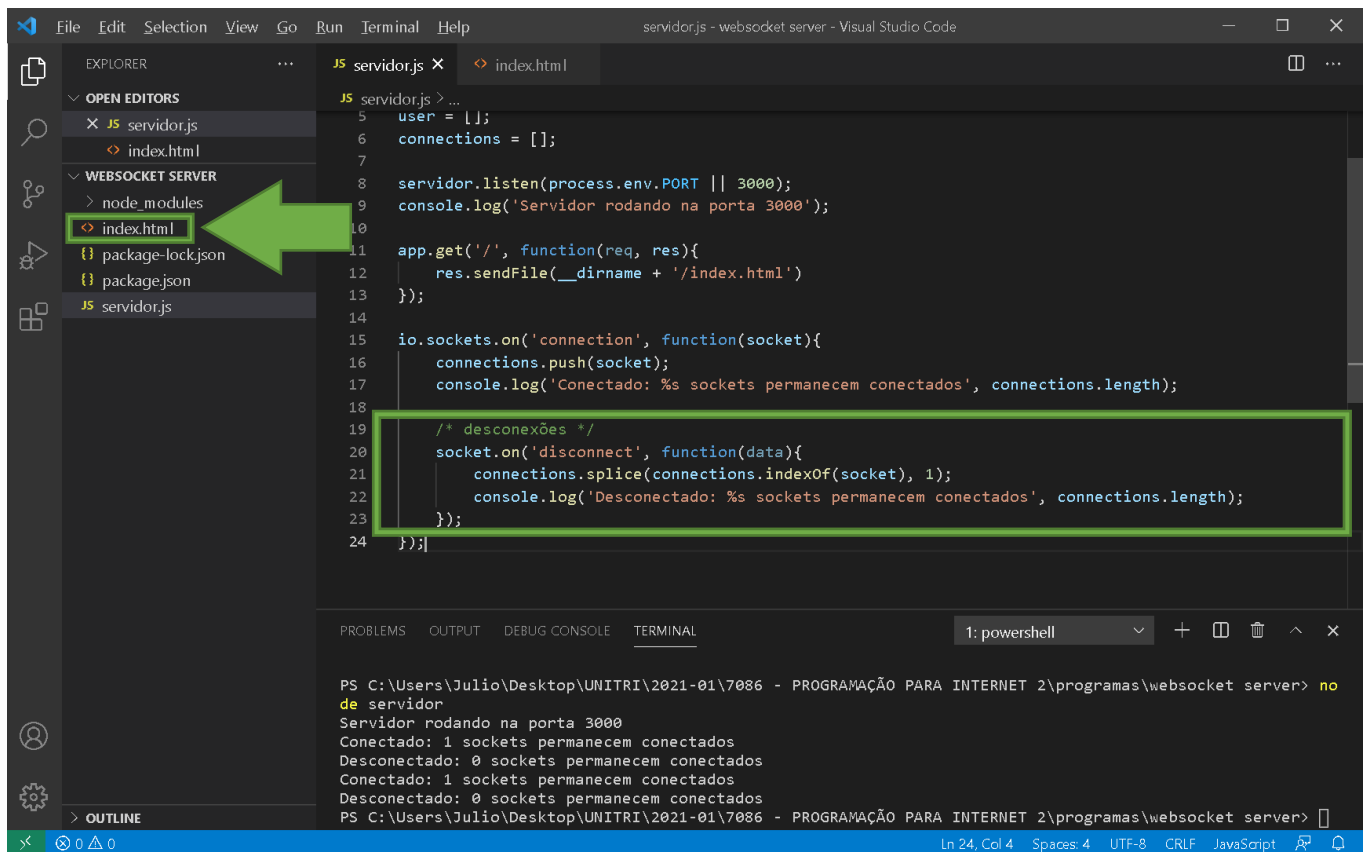
⇒ Ajuste o arquivo “index.html”, pare o servidor e rode novamente:



⇒ Atualize o navegador algumas vezes e observe os logs gerados pelo terminal:



⇒ Ajuste o arquivo “servidor.js”:



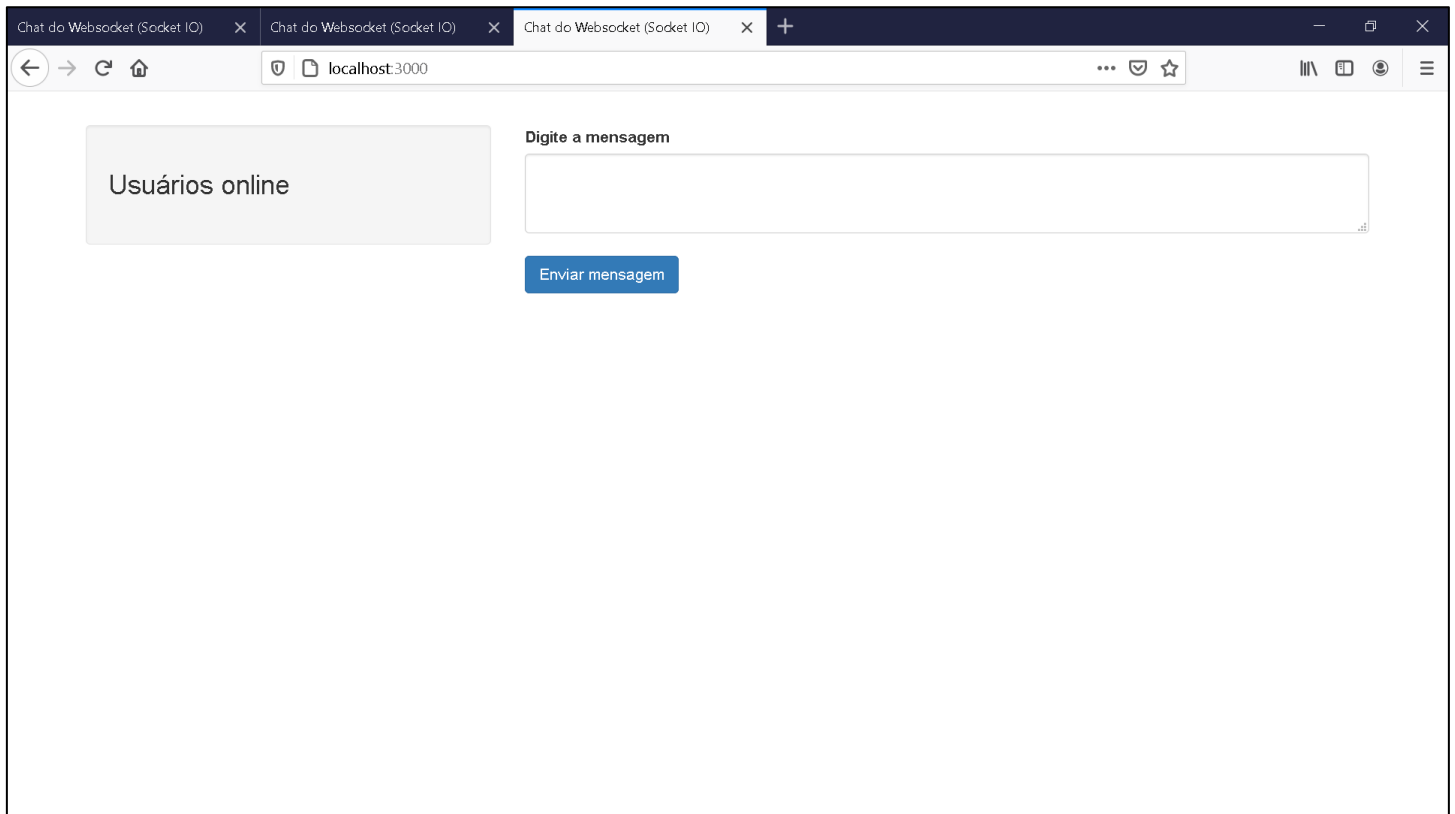
The screenshot shows the Visual Studio Code interface. On the left, the Explorer sidebar shows the project structure with 'index.html' highlighted under the 'WEBSOCKET SERVER' folder. A green arrow points to 'index.html'. The main editor displays the 'servidor.js' file with the following code:

```
JS servidor.js > ...
5  user = {};
6  connections = [];
7
8  servidor.listen(process.env.PORT || 3000);
9  console.log('Servidor rodando na porta 3000');
10
11 app.get('/', function(req, res){
12   res.sendFile(__dirname + '/index.html')
13 });
14
15 io.sockets.on('connection', function(socket){
16   connections.push(socket);
17   console.log('Conectado: %s sockets permanecem conectados', connections.length);
18
19   /* desconexões */
20   socket.on('disconnect', function(data){
21     connections.splice(connections.indexOf(socket), 1);
22     console.log('Desconectado: %s sockets permanecem conectados', connections.length);
23   });
24 });
```

The terminal window at the bottom shows the command 'node servidor' and the output:

```
PS C:\Users\Julio\Desktop\UNITRI\2021-01\7086 - PROGRAMAÇÃO PARA INTERNET 2\programas\websocket server> node
de servidor
Servidor rodando na porta 3000
Conectado: 1 sockets permanecem conectados
Desconectado: 0 sockets permanecem conectados
Conectado: 1 sockets permanecem conectados
Desconectado: 0 sockets permanecem conectados
PS C:\Users\Julio\Desktop\UNITRI\2021-01\7086 - PROGRAMAÇÃO PARA INTERNET 2\programas\websocket server>
```

⇒ Abra 3 abas no navegador com o “http://localhost:3000/”:



⇒ Observe os logs no terminal:

```
File Edit Selection View Go Run Terminal Help
servidor.js - websocket server - Visual Studio Code

EXPLORER
  OPEN EDITORS
    JS servidor.js
    index.html
  WEBSOCKET SERVER
    node_modules
    index.html
    package-lock.json
    package.json
    JS servidor.js

JS servidor.js
5 user = [];
6 connections = [];
7
8 servidor.listen(process.env.PORT || 3000);
9 console.log('Servidor rodando na porta 3000');
10
11 app.get('/', function(req, res){
12   res.sendFile(__dirname + '/index.html')
13 });
14
15 io.sockets.on('connection', function(socket){
16   connections.push(socket);
17   console.log('Conectado: %s sockets permanecem conectados', connections.length);
18
19   /* desconexões */
20   socket.on('disconnect', function(data){
21     connections.splice(connections.indexOf(socket), 1);
22     console.log('Desconectado: %s sockets permanecem conectados', connections.length);
23   });
24 });

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: node

Conectado: 1 sockets permanecem conectados
Desconectado: 0 sockets permanecem conectados
PS C:\Users\Julio\Desktop\UNITRI\2021-01\7086 - PROGRAMAÇÃO PARA INTERNET 2\programas\websocket server> node
de servidor
Servidor rodando na porta 3000
Conectado: 1 sockets permanecem conectados
Conectado: 2 sockets permanecem conectados
Conectado: 3 sockets permanecem conectados
```

⇒ Adicione o seguinte ajuste ao arquivo “servidor.js”:

```
File Edit Selection View Go Run Terminal Help
servidor.js - websocket server - Visual Studio Code

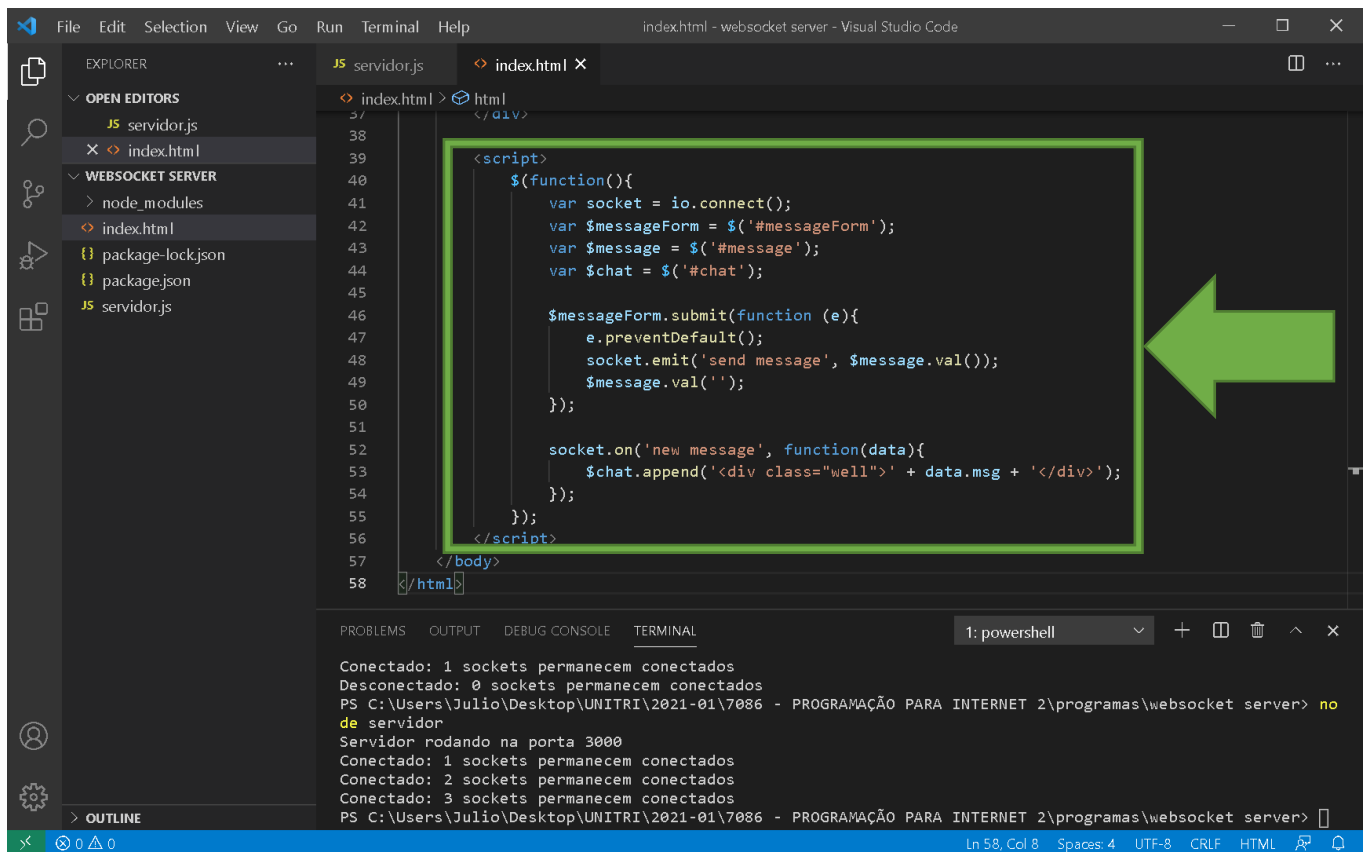
EXPLORER
  OPEN EDITORS
    JS servidor.js
    index.html
  WEBSOCKET SERVER
    node_modules
    index.html
    package-lock.json
    package.json
    JS servidor.js

JS servidor.js
8 servidor.listen(process.env.PORT || 3000);
9 console.log('Servidor rodando na porta 3000');
10
11 app.get('/', function(req, res){
12   res.sendFile(__dirname + '/index.html')
13 });
14
15 io.sockets.on('connection', function(socket){
16   connections.push(socket);
17   console.log('Conectado: %s sockets permanecem conectados', connections.length);
18
19   /* desconexões */
20   socket.on('disconnect', function(data){
21     connections.splice(connections.indexOf(socket), 1);
22     console.log('Desconectado: %s sockets permanecem conectados', connections.length);
23   });
24
25   /* envio de mensagens */
26   socket.on('send message', function(data){
27     io.sockets.emit('new message', {msg: data});
28   });
29 });

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: node

Conectado: 1 sockets permanecem conectados
Desconectado: 0 sockets permanecem conectados
PS C:\Users\Julio\Desktop\UNITRI\2021-01\7086 - PROGRAMAÇÃO PARA INTERNET 2\programas\websocket server> node
de servidor
Servidor rodando na porta 3000
Conectado: 1 sockets permanecem conectados
Conectado: 2 sockets permanecem conectados
Conectado: 3 sockets permanecem conectados
```

⇒ Ajuste o arquivo “index.html”:



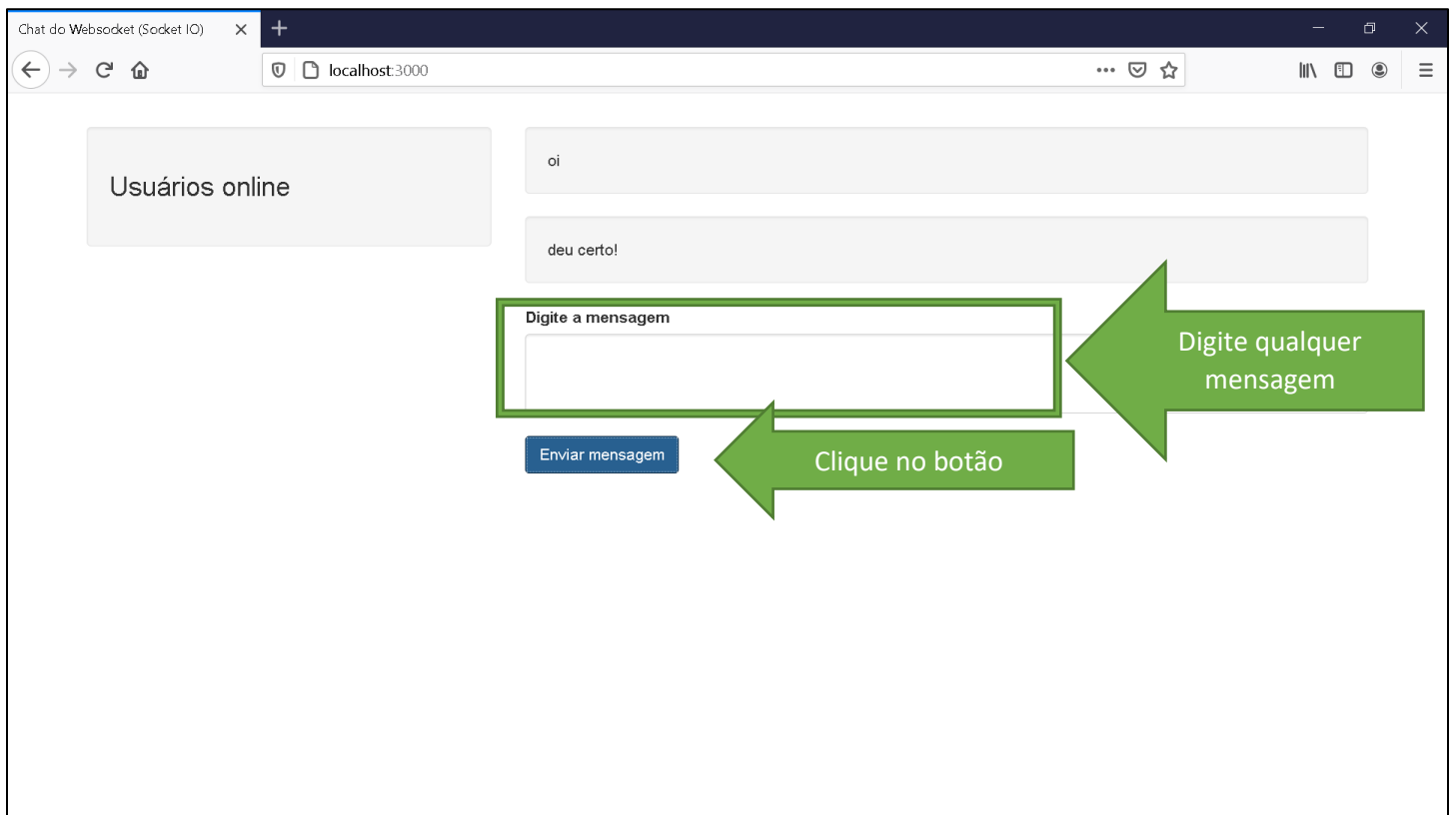
```
37 </div>
38
39
40 <script>
41     $(function(){
42         var socket = io.connect();
43         var $messageForm = $('#messageForm');
44         var $message = $('#message');
45         var $chat = $('#chat');
46
47         $messageForm.submit(function (e){
48             e.preventDefault();
49             socket.emit('send message', $message.val());
50             $message.val('');
51         });
52
53         socket.on('new message', function(data){
54             $chat.append('<div class="well"> ' + data.msg + '</div>');
55         });
56     });
57 </script>
58 </body>
59 </html>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

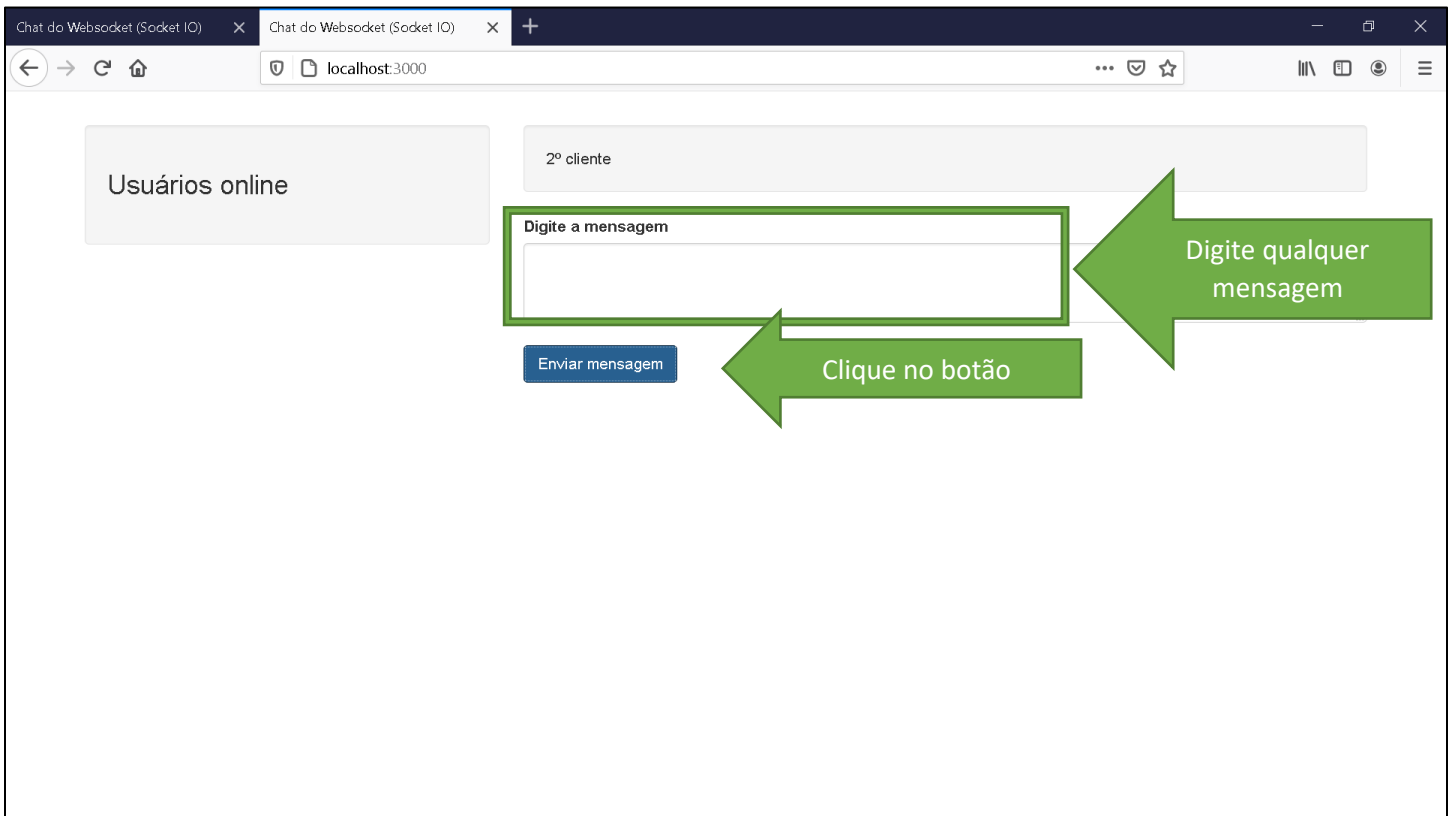
1: powershell

Conectado: 1 sockets permanecem conectados
Desconectado: 0 sockets permanecem conectados
PS C:\Users\Julio\Desktop\UNITRI\2021-01\7086 - PROGRAMAÇÃO PARA INTERNET 2\programas\websocket server> no
de servidor
Servidor rodando na porta 3000
Conectado: 1 sockets permanecem conectados
Conectado: 2 sockets permanecem conectados
Conectado: 3 sockets permanecem conectados
PS C:\Users\Julio\Desktop\UNITRI\2021-01\7086 - PROGRAMAÇÃO PARA INTERNET 2\programas\websocket server>

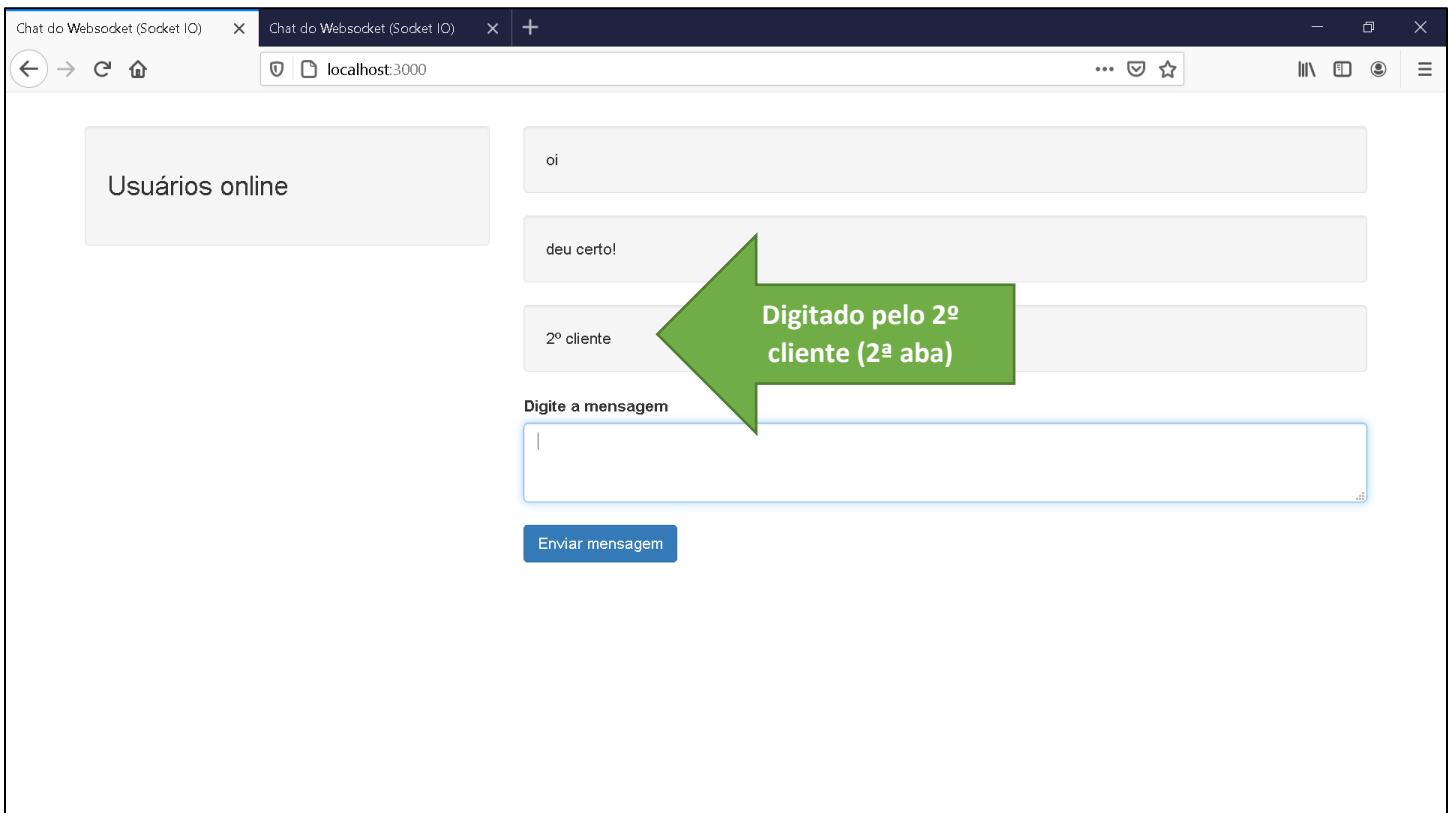
⇒ Atualize o navegador:



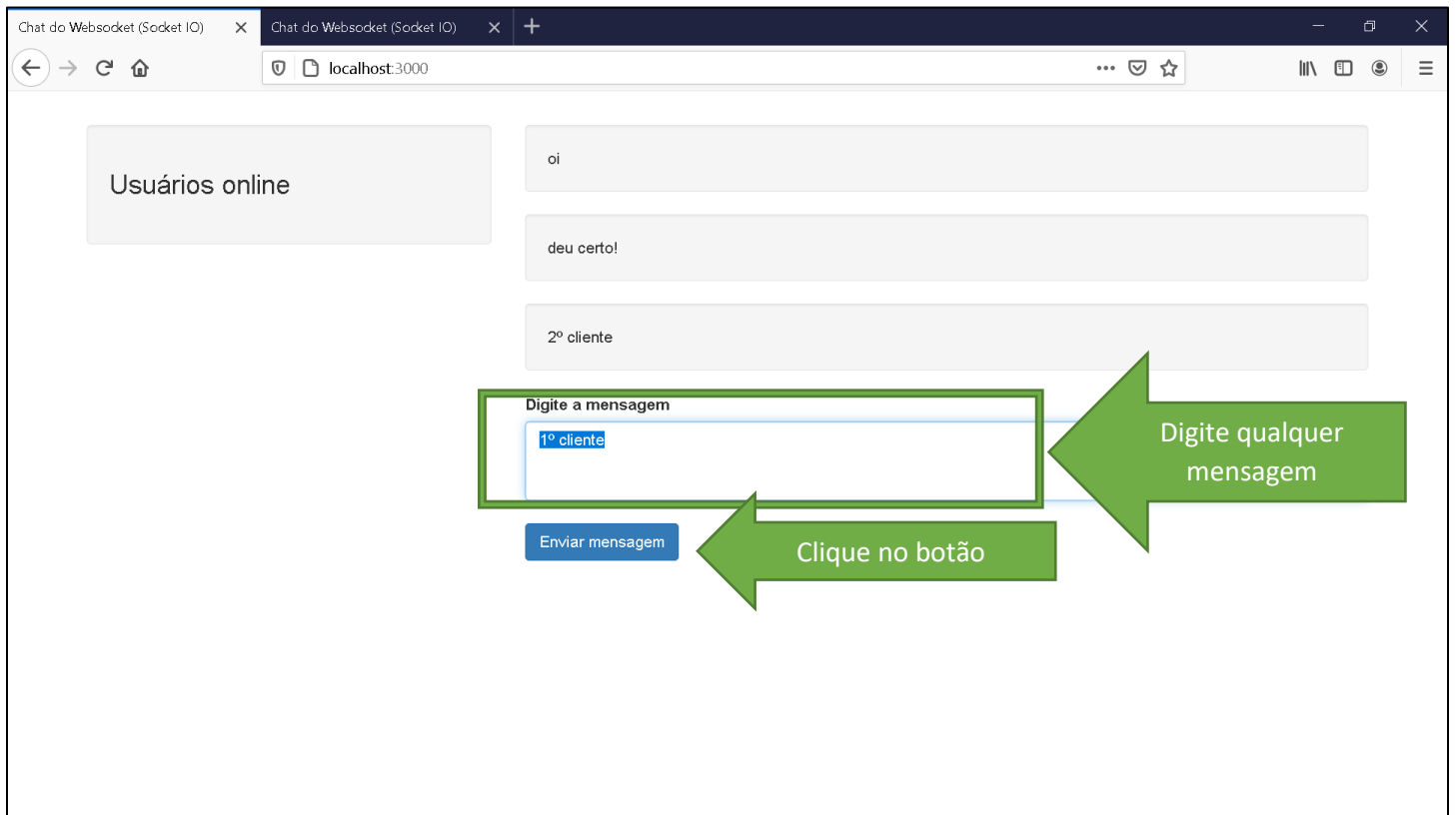
⇒ Abra outra aba do navegador:



⇒ Observe a digitação na outra aba do navegador:



⇒ Digite outra coisa na 1ª aba do navegador:



⇒ Observe a digitação na outra aba do navegador:

