



Backend

Prof. Renan Ponick



Lema

"A maldição do conhecimento é que ele fecha nossa mente para o que não sabemos."

Adam Grant



Agenda

30/10/2023 - Diagramas + Trabalho

06/11/2023 - Diagramas


13/11/2023 - Teste + TDD

20/11/2023 - BD

27/11/2023 - Debug

04/12/2023 - Apresentação

11/12/2023 - Avaliação + Recuperação



Bugs e Erros

Como identificá-los no meio de códigos gigantes?

Solução 1

```
escreva("entrou aqui");
```

```
echo "funciona";
```

```
printr("batata");
```

```
console.log("palavrão horrendo devido ao stress do  
bug não encontrado");
```

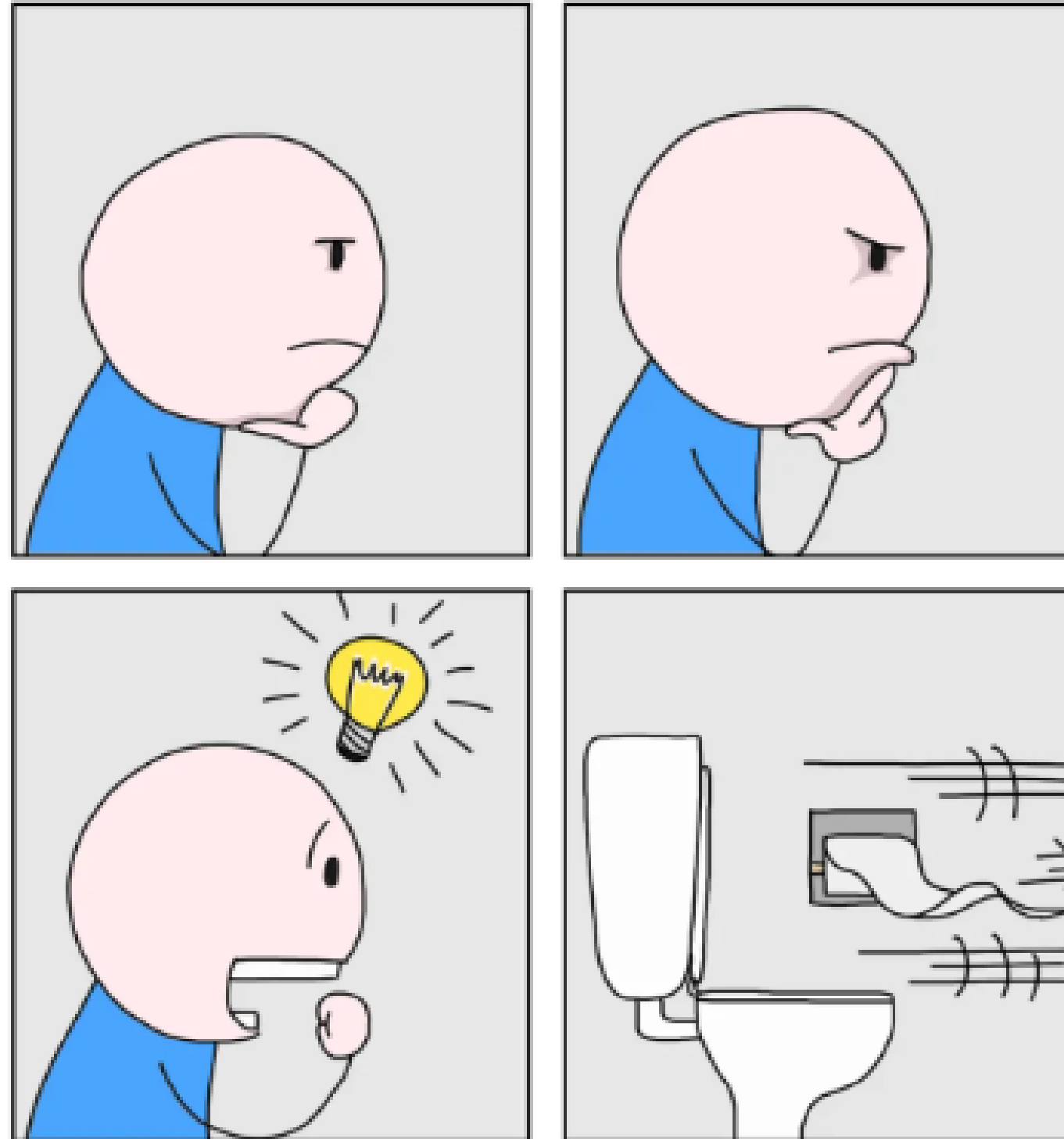


Solução 2

Debug ou Debugging
Depurar ou Depuração

Solução 2

DEBUGGING



Solução 2

The image shows a Visual Studio Code editor window with a Node.js application being debugged. The editor is open to the file `app.js`, which contains the following code:

```
2 const app = express();
3 const http = require('http').Server(app);
4 const io = require('socket.io')(http);
5 const path = require('path');
6
7 //Serve public directory
8 app.use(express.static(path.join(__dirname, 'public')));
9
10 app.get('/', function(req, res) {
11   res.sendFile(path.join(__dirname, 'public/index.html'));
12 });
13
14 io.on('connection', function(socket) {
15   console.log('a new user connected ');
16
17   socket.on('disconnect', () => {
18     console.log('user disconnected');
19   });
20 });
```

The `launch.json` file is also open, showing the configuration for the debug session:

```
{
  "version": "0.2.0",
  "configurations": [
    {
      "name": "Launch Program",
      "type": "node",
      "request": "launch",
      "program": "${workspaceFolder}/app.js",
      "args": [],
      "cwd": "${workspaceFolder}",
      "env": {},
      "externalConsole": false,
      "debugger": "node"
    }
  ]
}
```

The left sidebar shows the following panels:

- VARIABLES**: Shows the current scope (Local) with variables `socket` and `this`.
- WATCH**: Shows variables `usernameInput` and `messages` as `not available`.
- CALL STACK**: Shows the call stack with the current frame being `(anonymous function) app.js 15:2`.
- BREAKPOINTS**: Shows a breakpoint set at line 15 of `app.js`.

The bottom panel shows the **DEBUG CONSOLE** with the following output:

```
Debugging with legacy protocol because Node.js v7.10.0 was detected.
/usr/local/bin/node --nolazy --debug-brk=60065 app.js
(node:20173) DeprecationWarning: node --debug is deprecated. Please use node --inspect instead.
Debugger listening on 127.0.0.1:60065
listening on port 3000
```

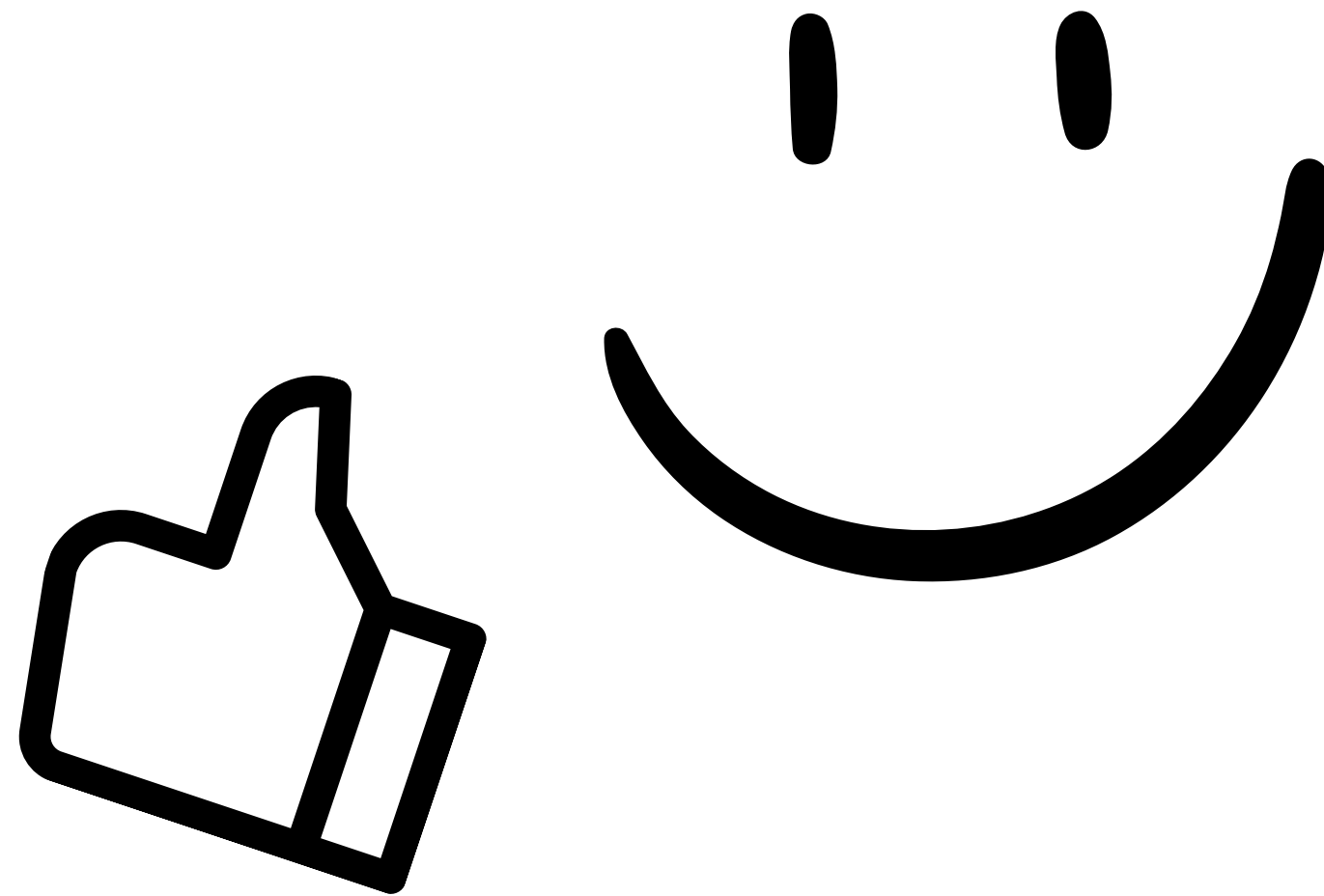



Definição

É o processo voltado para identificar e remover erros existentes no código.

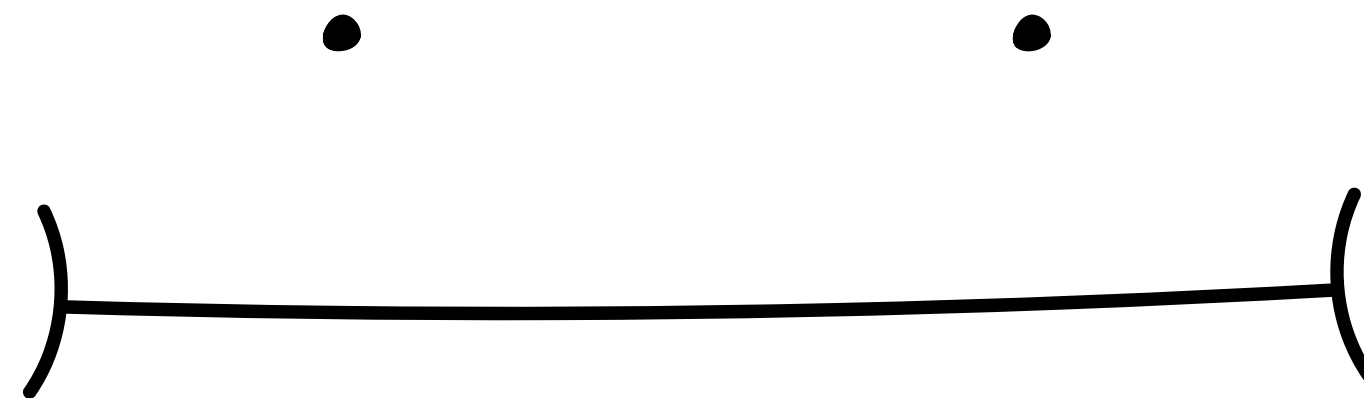
Dica

Ele não descobre erros sozinho!



Dica

Ele não analisa erros sozinho!



Dica

Ele não corrige erros sozinho!





e agora?

Você vai precisar reconhecer o bug, encontrar exatamente em qual parte do código ele esconde, para só então ativar o modo debug e DETALHAR o erro.

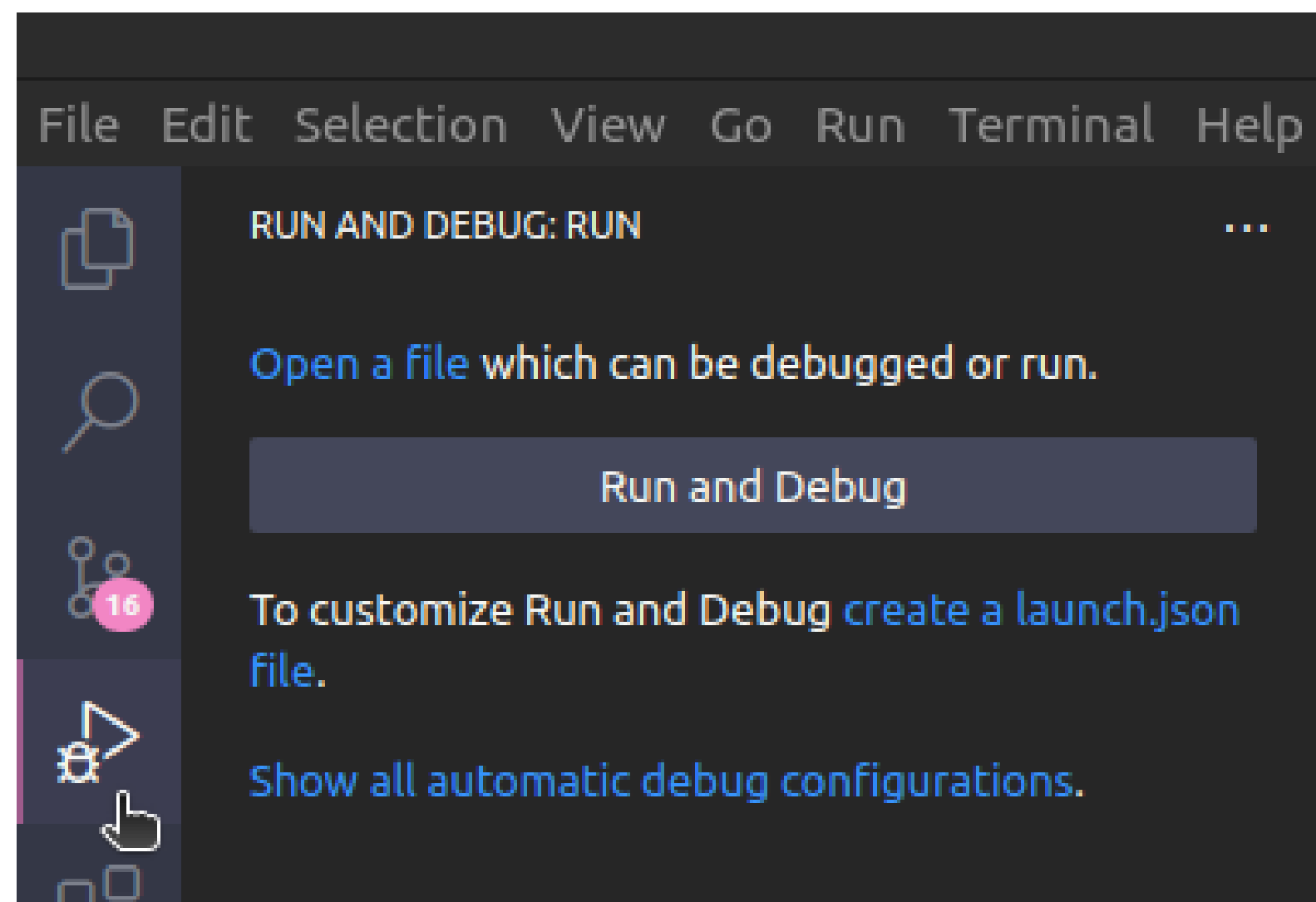


Importancia

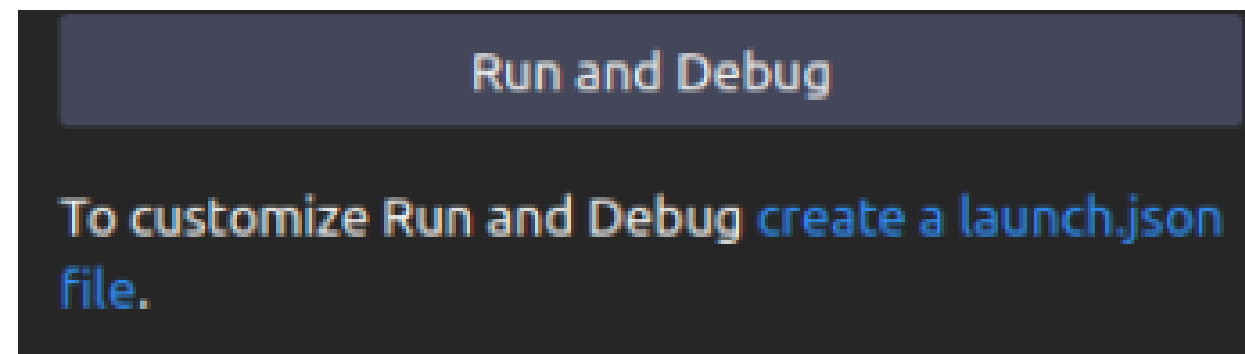
O debug melhora a gestão de tempo e produtividade. É nítido que corrigir bugs manualmente pode ser uma tarefa demorada e frustrante.

Na prática

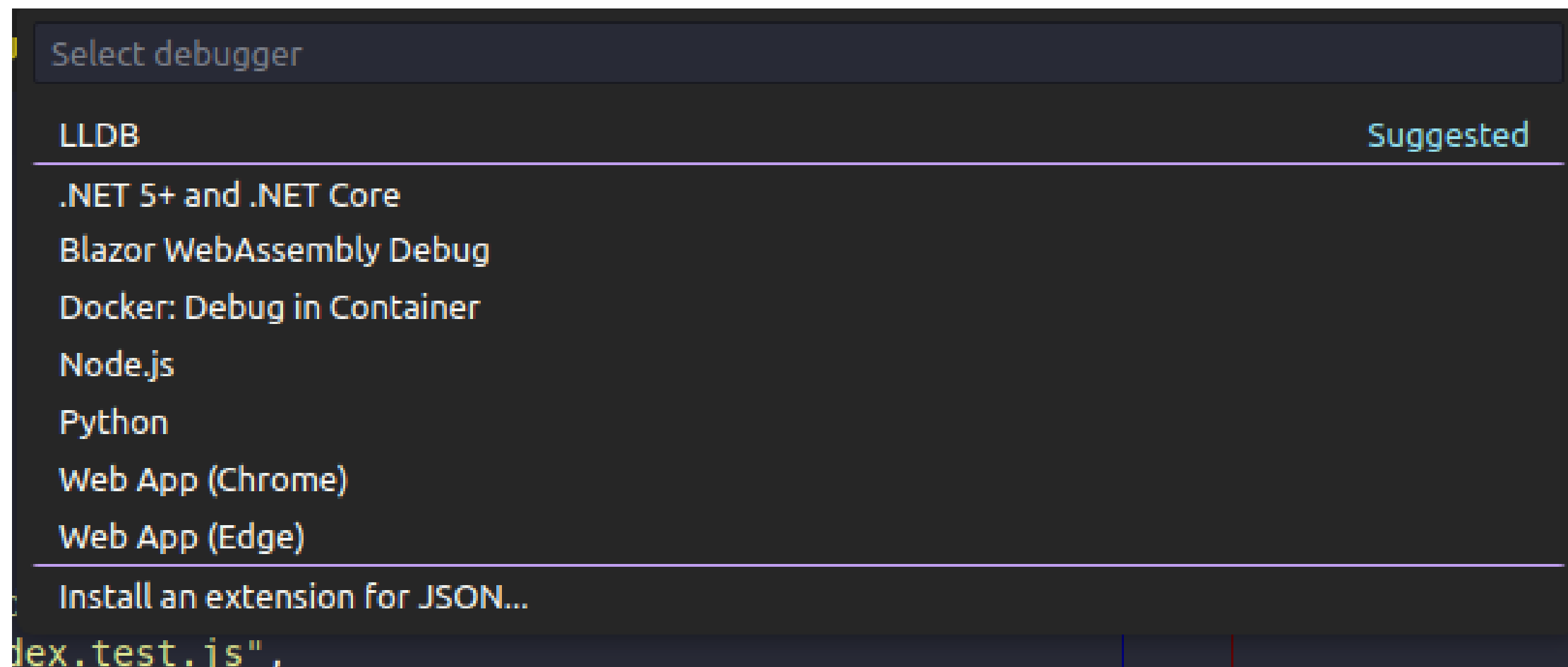
Dentro do VsCode encontre o icone do debug



Na prática



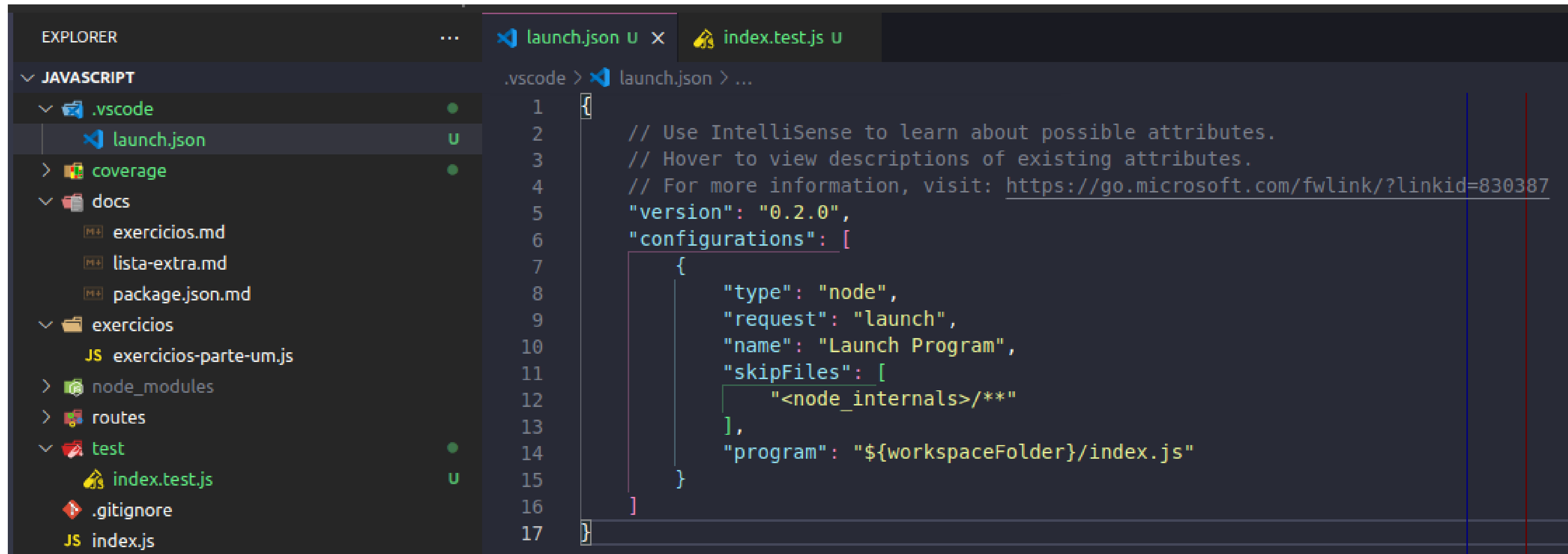
Abaixo do botão Run and Debug, clique na msg em azul



Selecione Node.js

Na prática

Será gerado um arquivo de configuração para debugar aplicações node:

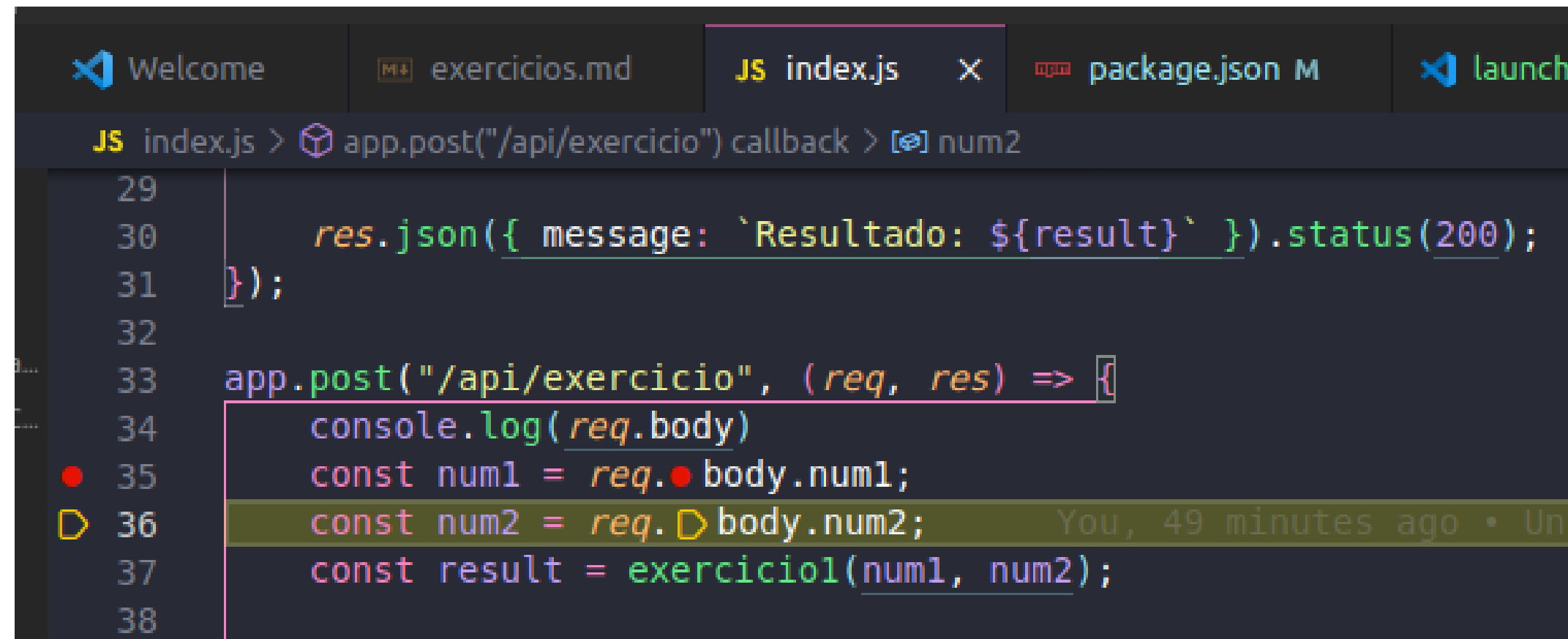


The screenshot shows the Visual Studio Code interface. On the left, the Explorer sidebar displays the file structure of a project, including folders like .vscode, coverage, docs, and test, and files like launch.json, index.test.js, and index.js. The main editor area shows the launch.json file, which is a JSON configuration for debugging Node.js applications. The file contains comments and a configuration for a 'Launch Program'.

```
1 {  
2   // Use IntelliSense to learn about possible attributes.  
3   // Hover to view descriptions of existing attributes.  
4   // For more information, visit: https://go.microsoft.com/fwlink/?linkid=830387  
5   "version": "0.2.0",  
6   "configurations": [  
7     {  
8       "type": "node",  
9       "request": "launch",  
10      "name": "Launch Program",  
11      "skipFiles": [  
12        "<node_internals>/**"  
13      ],  
14      "program": "${workspaceFolder}/index.js"  
15    }  
16  ]  
17 }
```

Na prática

Adicione breakpoints no código, faça uma requisição e veja a mágica acontecer.

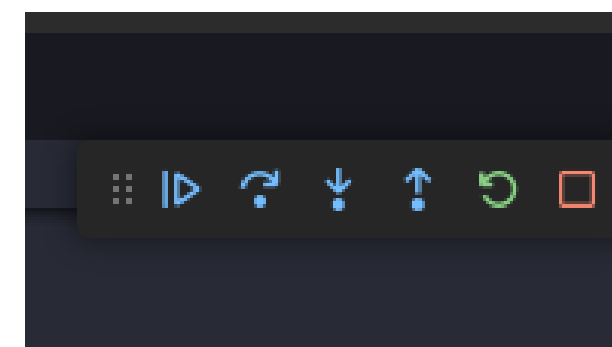


The screenshot shows the Visual Studio Code editor with a JavaScript file named `index.js`. The code is as follows:

```
29  
30     res.json({ message: `Resultado: ${result}` }).status(200);  
31 });  
32  
33 app.post("/api/exercicio", (req, res) => {  
34     console.log(req.body)  
35     const num1 = req.body.num1;  
36     const num2 = req.body.num2;  
37     const result = exercicio1(num1, num2);  
38
```

A red dot breakpoint is set on line 36. The breadcrumb at the top indicates the current position: `JS index.js > app.post("/api/exercicio") callback > num2`. The file explorer on the left shows `exercicios.md`, `index.js`, `package.json`, and `launch`.

Esses são os controladores do debug





Exercícios

Corrija os outros 3 erros do projeto que envie