

# Подигнување на работна околина

-вежби-



ФАКУЛТЕТ ЗА ЕЛЕКТРОТЕХНИКА И  
ИНФОРМАЦИСКИ ТЕХНОЛОГИИ

FACULTY OF ELECTRICAL ENGINEERING  
AND INFORMATION TECHNOLOGIES

# Тамара Митевска



❖ E-mail:

[tamara\\_mitevska@hotmail.com](mailto:tamara_mitevska@hotmail.com)

❖ LinkedIn:

<https://www.linkedin.com/in/tamara-mitevska-988737197/>

❖ Github:

<https://github.com/TMitevska>





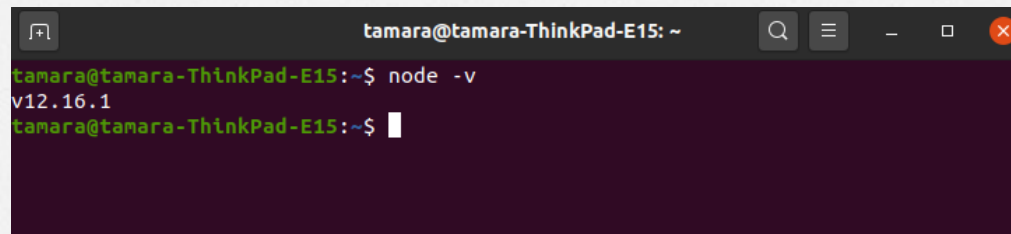
# Инсталирање

Инсталирање со команда:

- ☐ **sudo apt-get update**
- ☐ **sudo apt-get dist-upgrade**
- ☐ **curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.35.2/install.sh | bash**
- ☐ **source ~/.bashrc**
- ☐ **nvm install lts/erbium**

Верзии

<https://nodejs.org/en/download/releases/>

A terminal window with a dark purple background. The title bar shows 'tamara@tamara-ThinkPad-E15: ~'. The terminal text shows the command 'node -v' being executed, resulting in the output 'v12.16.1'. The prompt 'tamara@tamara-ThinkPad-E15:~\$' is visible at the end of the line.

```
tamara@tamara-ThinkPad-E15: ~  
tamara@tamara-ThinkPad-E15:~$ node -v  
v12.16.1  
tamara@tamara-ThinkPad-E15:~$
```

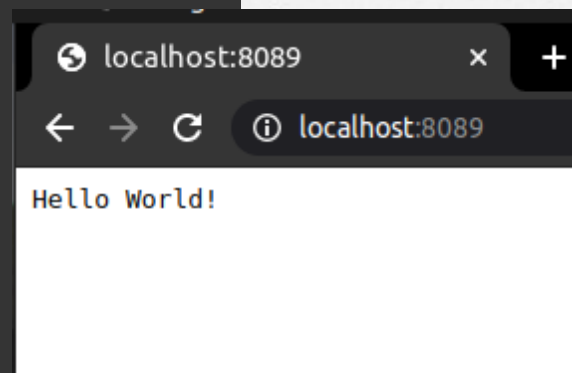
# Подигнување на локален сервер

```
Open ▼ [⌘]
const http = require('http')

const hostname = '127.0.0.1'
const port = 8089

const server = http.createServer((req, res) => {
  res.statusCode = 200
  res.setHeader('Content-Type', 'text/plain')
  res.end('Hello World!\n')
})

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`)
})
```



```
tamara@tamara-ThinkPad-E15: ~/Desktop
tamara@tamara-ThinkPad-E15:~$ node -v
v12.16.1
tamara@tamara-ThinkPad-E15:~$ cd Desktop/
tamara@tamara-ThinkPad-E15:~/Desktop$ node raspberryTest.js
Server running at http://127.0.0.1:8089/
```



# Креирање на сервер со Express

- ☐ `mkdir expressServer`
- ☐ `cd expressServer`
- ☐ `touch raspberryServerExpress.js`
- ☐ `npm init`
- ☐ `npm install express --save`
- ☐ `npm install --save body-parser`

```
Open  ▾  [⌕]
```

```
const express = require('express')
const app = express()
const port = 3000

app.get('/', (req, res) => {
  res.send('Hello World!')
  console.log('get method called!')
})

app.listen(port, () => {
  console.log(`Example app listening at http://localhost:${port}`)
})
```

```
tamara@tamara-ThinkPad-E15:~/Desktop/raspberryCode$ npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.
```

See `npm help json` for definitive documentation on these fields and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and save it as a dependency in the package.json file.

Press ^C at any time to quit.

package name: (raspberrycode)

version: (1.0.0)

description:

entry point: (raspberryServer.js)

test command:

git repository:

keywords:

author:

license: (ISC)

About to write to /home/tamara/Desktop/raspberryCode/package.json:

```
{
  "name": "raspberrycode",
  "version": "1.0.0",
  "description": "",
  "main": "raspberryServer.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}
```

Is this OK? (yes)

```
New minor version of npm available! 6.13.4 → 6.14.8
Changelog: https://github.com/npm/cli/releases/tag/v6.14.8
Run npm install -g npm to update!
```

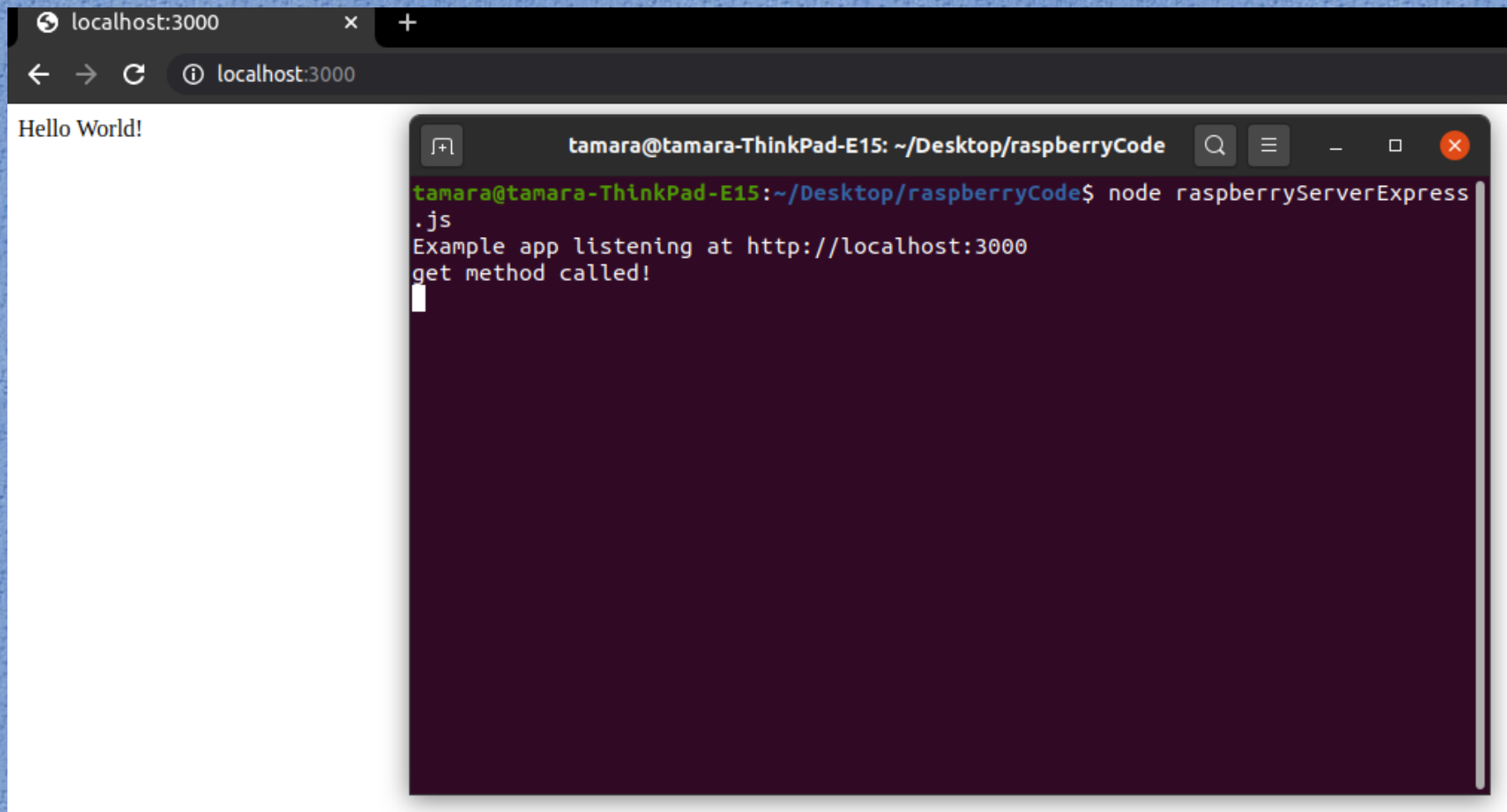
```
tamara@tamara-ThinkPad-E15:~/Desktop/raspberryCode$ npm install express --save
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN raspberrycode@1.0.0 No description
npm WARN raspberrycode@1.0.0 No repository field.
```

```
+ express@4.17.1
added 50 packages from 37 contributors and audited 50 packages in 2.186s
found 0 vulnerabilities
```

```
New minor version of npm available! 6.13.4 → 6.14.8
Changelog: https://github.com/npm/cli/releases/tag/v6.14.8
Run npm install -g npm to update!
```

```
tamara@tamara-ThinkPad-E15:~/Desktop/raspberryCode$ node raspberryServerExpress.js
Example app listening at http://localhost:3000
```





# Задача 1:

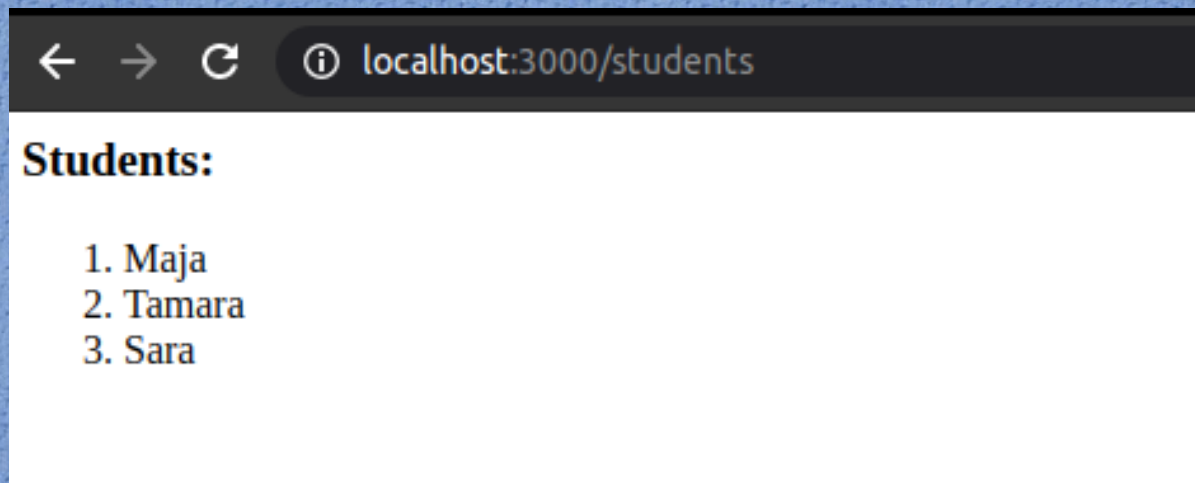
Креирај node.js express сервер на raspberry pi кој ќе може да се справи со следниве барања:

- ☐ /students – GET метод кој ќе врати HTML документ со листа од студенти
- ☐ /students – POST метод кој ќе се користи за додавање на нови студенти (се проследува само име на студент како параметар)
- ☐ /students – PUT метод кој ќе се користи за изменување на веќе постоечки студент (се проследува претходно и ново име на студент како параметри)
- ☐ /students – DELETE метод кој ќе се користи за бришење на студент (се проследува само име на студент како параметар)



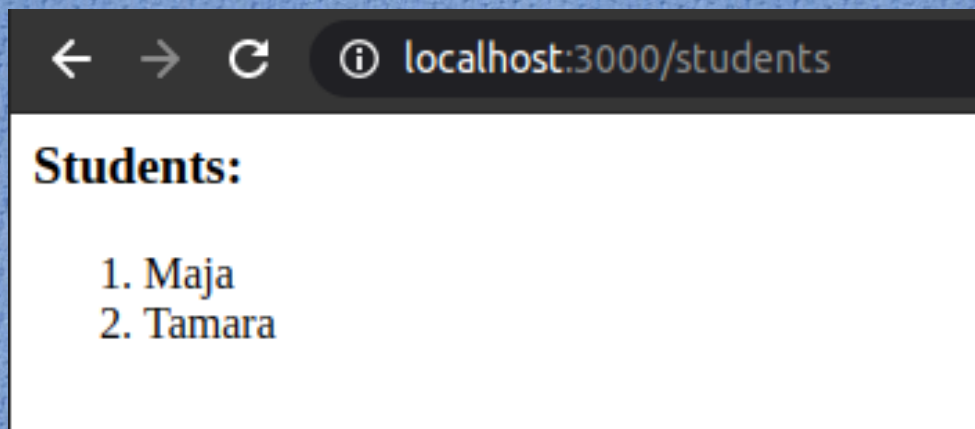
```
tamara@tamara-ThinkPad-E15: ~/Desktop/raspberryCode
tamara@tamara-ThinkPad-E15:~/Desktop/raspberryCode$ node httpMethods.js
Example app listening at http://localhost:3000

tamara@tamara-ThinkPad-E15:~/Desktop/raspberryCode$ curl -d '{"name":"Maja"}' -H "Content-Type: application/json" -X POST http://localhost:3000/students
Created new student Maja
tamara@tamara-ThinkPad-E15:~/Desktop/raspberryCode$ curl -d '{"name":"Tamara"}' -H "Content-Type: application/json" -X POST http://localhost:3000/students
Created new student Tamaratamara@tamara-ThinkPad-E15:~/Desktop/raspberryCode$ curl -d '{"name":"Sara"}' -H "Content-Type: application/json" -X POST http://localhost:3000/students
Created new student Sara
tamara@tamara-ThinkPad-E15:~/Desktop/raspberryCode$
```



```
tamara@tamara-ThinkPad-E15: ~/Desktop/raspberryCode
tamara@tamara-ThinkPad-E15:~/Desktop/raspberryCode$ node httpMethods.js
Example app listening at http://localhost:3000

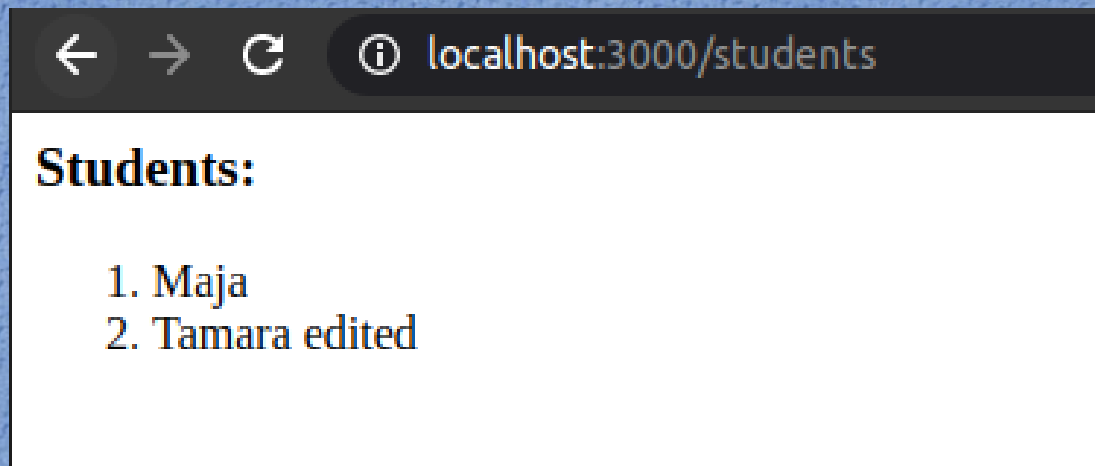
tamara@tamara-ThinkPad-E15:~/Desktop/raspberryCode$ curl -d '{"name":"Maja"}' -H "Content-Type: application/json" -X POST http://localhost:3000/students
Created new student Maja
tamara@tamara-ThinkPad-E15:~/Desktop/raspberryCode$ curl -d '{"name":"Tamara"}' -H "Content-Type: application/json" -X POST http://localhost:3000/students
Created new student Tamara
tamara@tamara-ThinkPad-E15:~/Desktop/raspberryCode$ curl -d '{"name":"Sara"}' -H "Content-Type: application/json" -X POST http://localhost:3000/students
Created new student Sara
tamara@tamara-ThinkPad-E15:~/Desktop/raspberryCode$ curl -d '{"name":"Sara"}' -H "Content-Type: application/json" -X DELETE http://localhost:3000/students
Deleted student Sara
tamara@tamara-ThinkPad-E15:~/Desktop/raspberryCode$
```





```
tamara@tamara-ThinkPad-E15: ~/Desktop/raspberrycode
tamara@tamara-ThinkPad-E15:~/Desktop/raspberrycode$ node httpMethods.js
Example app listening at http://localhost:3000

tamara@tamara-ThinkPad-E15:~/Desktop/raspberrycode$ curl -d '{"name":"Maja"}' -H "Content-Type: application/json" -X POST http://localhost:3000/students
Created new student Maja
tamara@tamara-ThinkPad-E15:~/Desktop/raspberrycode$ curl -d '{"name":"Tamara"}' -H "Content-Type: application/json" -X POST http://localhost:3000/students
Created new student Tamara
tamara@tamara-ThinkPad-E15:~/Desktop/raspberrycode$ curl -d '{"name":"Sara"}' -H "Content-Type: application/json" -X POST http://localhost:3000/students
Created new student Sara
tamara@tamara-ThinkPad-E15:~/Desktop/raspberrycode$ curl -d '{"name":"Sara"}' -H "Content-Type: application/json" -X DELETE http://localhost:3000/students
Deleted student Sara
tamara@tamara-ThinkPad-E15:~/Desktop/raspberrycode$ curl -d '{"name":"Tamara","nameAfter":"Tamara edited"}' -H "Content-Type: application/json" -X PUT http://localhost:3000/students
Edited student Tamara
tamara@tamara-ThinkPad-E15:~/Desktop/raspberrycode$
```



# Решение:

```
Open ▼ ↗
1 const express = require('express')
2 const bodyParser = require('body-parser');
3
4 const app = express()
5 const port = 3000
6
7 app.use(bodyParser.json()); // support json encoded bodies
8 app.use(bodyParser.urlencoded({ extended: true })); // support encoded bodies
9
10 let students = []
11
12 app.route('/students')
13 .get(function (req, res) {
14   res.writeHead(200, { 'Content-Type': 'text/html' })
15   var data = '<html><body>'
16   if(students) {
17     data+='\<h3>Students:</h3><ol>'
18     students.forEach(student => data+='\<li>' + student + '\</li>')
19     data+='\</ol>'
20   }
21   res.write(data + '\</body></html>')
22   res.end()
23 })
24 .post(function (req, res) {
25   const student = req.body.name
26   students.push(student)
27   res.send('Created new student ' + student)
28   res.end()
29 })
30 .put(function (req, res) {
31   const name = req.body.name
32   const studentNameAfter = req.body.nameAfter
33   students.forEach(function(item, i) { if (item === name) students[i] = studentNameAfter })
34   res.send('Edited student ' + name)
35   res.end()
36 })
37 .delete(function (req, res) {
38   const student = req.body.name
39   students = students.filter(v => v !== student);
40   res.send('Deleted student ' + student)
41   res.end()
42 })
43
44 app.listen(port, () => {
45   console.log(`Example app listening at http://localhost:${port}`)
46 })
```

Тамара Митевска

Факултет за електротехника и информациски технологии



## Задача 2

Креирај node.js express сервер на raspberry pi кој ќе може да се справи со следниве барања:

- ☐ /courses – GET метод кој ќе врати HTML документ со листа од курсеви
- ☐ /courses – POST метод кој ќе се користи за додавање на нови курсеви (се проследува само име на курс како параметар)
- ☐ /courses – PUT метод кој ќе се користи за изменување на веќе постоечки курс (се проследува претходно и ново име на курс како параметри)
- ☐ /courses – DELETE метод кој ќе се користи за бришење на курс (се проследува само име на курс како параметар)
- ☐ /students – GET метод кој ќе врати HTML документ со листа од студенти
- ☐ /students – POST метод кој ќе се користи за додавање на нови студенти (се проследува име на студент и име на курс како параметри)

# Задача 3

Креирај node.js express сервер на raspberry pi кој ќе може да се справи со следниве барања:

- ☐ /courses – GET метод кој ќе врати HTML документ со листа од курсеви
- ☐ /courses/:id – GET метод кој ќе го врати бараниот курс
- ☐ /courses – POST метод кој ќе се користи за додавање на нови курсеви (се проследува id, име на курс, почетен и краен датум како параметри)
- ☐ /courses/:id – PUT метод кој ќе се користи за изменување на веќе постоечки курс (се проследува име на курс, почетен и краен датум како патаметар)
- ☐ /courses/:id – DELETE метод кој ќе се користи за бришење на курс



*Прашања?*

**БЛАГОДАРАМ НА  
ВНИМАНИЕТО! 😊**