

# Градење на паметна соба или уред

- вежби -



Паметните компоненти играат централна улога во дизајнот на еден систем. Опремени со современа информатичка и комуникациска технологија, овие компоненти можат да складираат податоци, да се поврзуваат меѓусебно, да пристапат до сервиси и да комуницираат еден со друг.

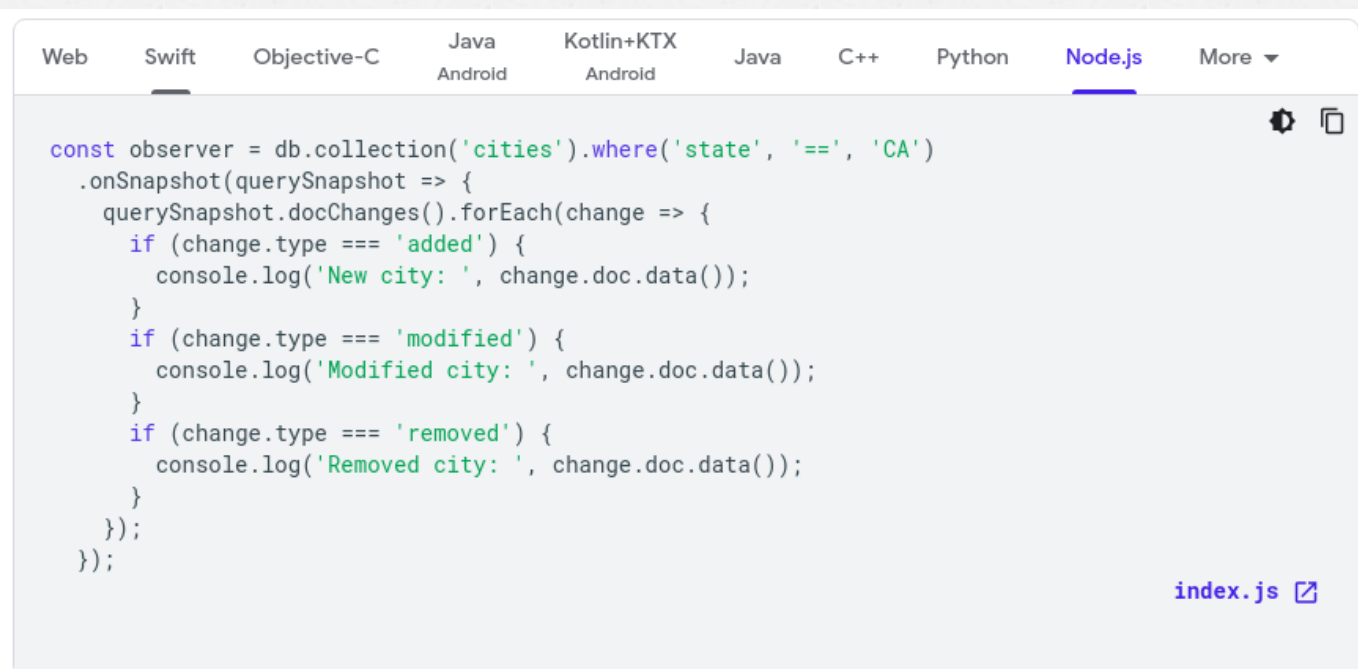
Сите компании бараат систем кој ќе биде реалистичен, достапен и едноставен. Во денешните услови, паметниот систем треба да биде едноставен, бидејќи корисниците вообичаено не сакаат многу промени. Достапноста е исто така особено важен фактор, бидејќи не секоја компанија е во можност да плати голема сума за некој паметен систем. Користење на најновите и најскапи технологии е посакувано, но понекогаш и подостапни компоненти може да дадат исти или слични перформанси. Од голема важност е системот да биде реалистичен, односно да може да се стави во употреба за краток период. Секоја компанија не е подготвена да потроши многу време на имплементација на некој систем, дури и ако се работи за паметен систем.



# Промена на документ од база

Во апликации каде што е важно да се дојде до промени во реално време потребно е да се најде начин корисникот да се извести за промената откако истата ќе се сочува во база со податоци.

Cloud Firestore нуди посебен интерфејс за известување на серверот/корисникот кога ќе се направи некоја промена во документите.



```
Web Swift Objective-C Java Android Kotlin+KTX Android Java C++ Python Node.js More ▾  
  
const observer = db.collection('cities').where('state', '==', 'CA')  
.onSnapshot(querySnapshot => {  
  querySnapshot.docChanges().forEach(change => {  
    if (change.type === 'added') {  
      console.log('New city: ', change.doc.data());  
    }  
    if (change.type === 'modified') {  
      console.log('Modified city: ', change.doc.data());  
    }  
    if (change.type === 'removed') {  
      console.log('Removed city: ', change.doc.data());  
    }  
  });  
});  
  
index.js
```

# Задача 1

Креирај node.js express сервер на raspberry pi кој податоците ќе ги сочувува во Cloud Firestore база на податоци и ќе може да се справи со следниве барања:

- ❑ /home – GET метод кое враќа страница со 2 копчиња. Првото копче е Register device кое служи за додавање на нови паметни уреди. Како влезни параметри потребно е да се внесат име на уред и име на корисник кој го регистрира (може да се користи prompt) и истите да се проследат на POST метод /devices. Второто копче е Preview device status (real time) и при клик на истото се повикува GET метод /devices кое префрлува на друга страница.
- ❑ /devices – POST метод на кој му се проследува објект со име на уред, корисник, датум и статус вклучено/исклучено
- ❑ /devices – GET метод кој ќе врати страница со листа од сите регистрирани уреди и листа со историја на додадени, изменети или избришани уреди. Историја треба да ја сочувува секоја промена која ќе биде направена во табелата за уреди на облак



localhost:3000/home

Register device

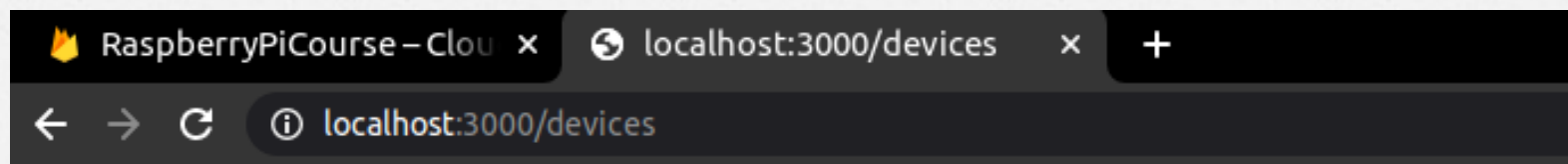
Preview device status (real time)

localhost:3000 says  
Please enter device name:

Living room light1

Cancel OK

<a href="#">🏠</a> > <a href="#">devices</a> > 1D6AikSgoDvSs... <a href="#">✎</a>		
<a href="#">🏠</a> raspberrypicourse	<a href="#">📁</a> devices	<a href="#">📄</a> 1D6AikSgoDvSsgezaBd1
<a href="#">+</a> Start collection	<a href="#">+</a> Add document	<a href="#">+</a> Start collection
courses	1D6AikSgoDvSsgezaBd1 >	<a href="#">+</a> Add field
<b>devices</b> >	eVnfFDfh0pLmr20tYT0r	date: "Tue Oct 06 2020"
students		device: "Living room light 1 edited"
		status: "on"
		user: "Tamara Mitevska"



### Registeres Devices:

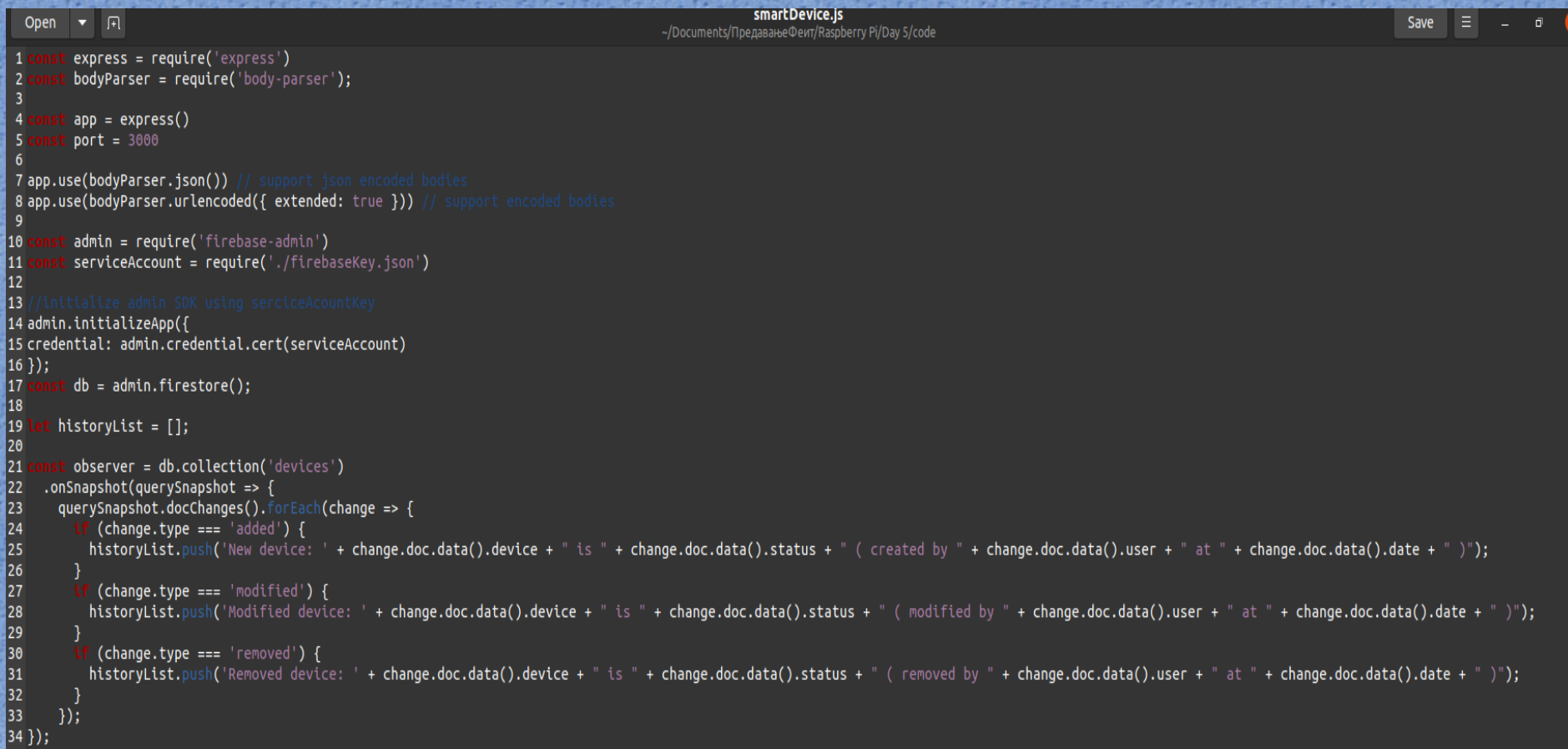
1. 1D6AikSgoDvSsgezaBd1 --> Living room light 1 edited is on
2. eVnfFDfh0pLmr20tYT0r --> Living room light 2 is on

### History:

1. New device: Living room light 1 edited is on ( created by Tamara Mitevaska at Tue Oct 06 2020 )
2. New device: Living room light 2 is on ( created by Tamara Mitevaska at Tue Oct 06 2020 )



# Решение



```
smartDevice.js
~/Documents/ПредаванеФеври/Рaspberry Pi/Day 5/code

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 const admin = require('firebase-admin')
11 const serviceAccount = require('./firebaseKey.json')
12
13 //initialize admin SDK using serviceAccountKey
14 admin.initializeApp({
15   credential: admin.credential.cert(serviceAccount)
16 });
17 const db = admin.firestore();
18
19 let historyList = [];
20
21 const observer = db.collection('devices')
22   .onSnapshot(querySnapshot => {
23     querySnapshot.docChanges().forEach(change => {
24       if (change.type === 'added') {
25         historyList.push('New device: ' + change.doc.data().device + " is " + change.doc.data().status + " ( created by " + change.doc.data().user + " at " + change.doc.data().date + " )");
26       }
27       if (change.type === 'modified') {
28         historyList.push('Modified device: ' + change.doc.data().device + " is " + change.doc.data().status + " ( modified by " + change.doc.data().user + " at " + change.doc.data().date + " )");
29       }
30       if (change.type === 'removed') {
31         historyList.push('Removed device: ' + change.doc.data().device + " is " + change.doc.data().status + " ( removed by " + change.doc.data().user + " at " + change.doc.data().date + " )");
32       }
33     });
34   });
```

Тамара Митевска

Факултет за електротехника и информациски технологии

```

36 app.get( "/home", (req, res) => {
37   res.writeHead(200, { 'Content-Type': 'text/html' })
38   var data = '<html><body>'
39
40   data += '<head><script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script><script>function
   getDeviceInfo() {const device = prompt("Please enter device name:", "Living room light1"); const user = prompt("Please enter
   user name:", "Tamara Mitevska"); const status = "on"; const date = new Date().toDateString(); const deviceInfo = {device, user,
   status, date};$.post("http://localhost:3000/devices", deviceInfo, function(result){alert("Device is registered!");});});
   function preview(){window.location="http://localhost:3000/devices";}</script></head>'
41
42   data += '<button onclick="getDeviceInfo()">Register device</button><br><br>'
43
44   data += '<button onclick="preview()">Preview device status (real time)</button>'
45
46   res.write(data + '</body></html>')
47   res.end()
48 })
49
50 app.get( "/devices", (req, res) => {
51   res.writeHead(200, { 'Content-Type': 'text/html' })
52   var data = '<html><body>'
53
54   db.collection('devices').get().then(doc => {
55     if(doc) {
56       data+='\<h3>Registeres Devices:</h3><ol>'
57       doc.forEach(device => data+='\<li>' + device.id + ' --> ' + device.data().device + " is " + device.data().status + '</-
   li>')
58       data+='\</ol>'
59     }
60     data += '\<h3>History:</h3><ol>'
61     historyList.forEach(item => data += '\<li>' + item + '</li>')
62     res.write(data + '</ol></body></html>')
63     res.end()
64   });
65 })
66
67 app.post( "/devices", (req, res) => {
68   const deviceInfo = {device: req.body.device, user: req.body.user, status: req.body.status, date: req.body.date}
69   db.collection('devices').add(deviceInfo).then(() => {
70     res.send('Created new device ' + deviceInfo.device)
71     res.end()
72   });
73 })
74
75 app.listen(port, () => {
76   console.log(`Example app listening at http://localhost:${port}`)
77 })

```

Тамара Митевска

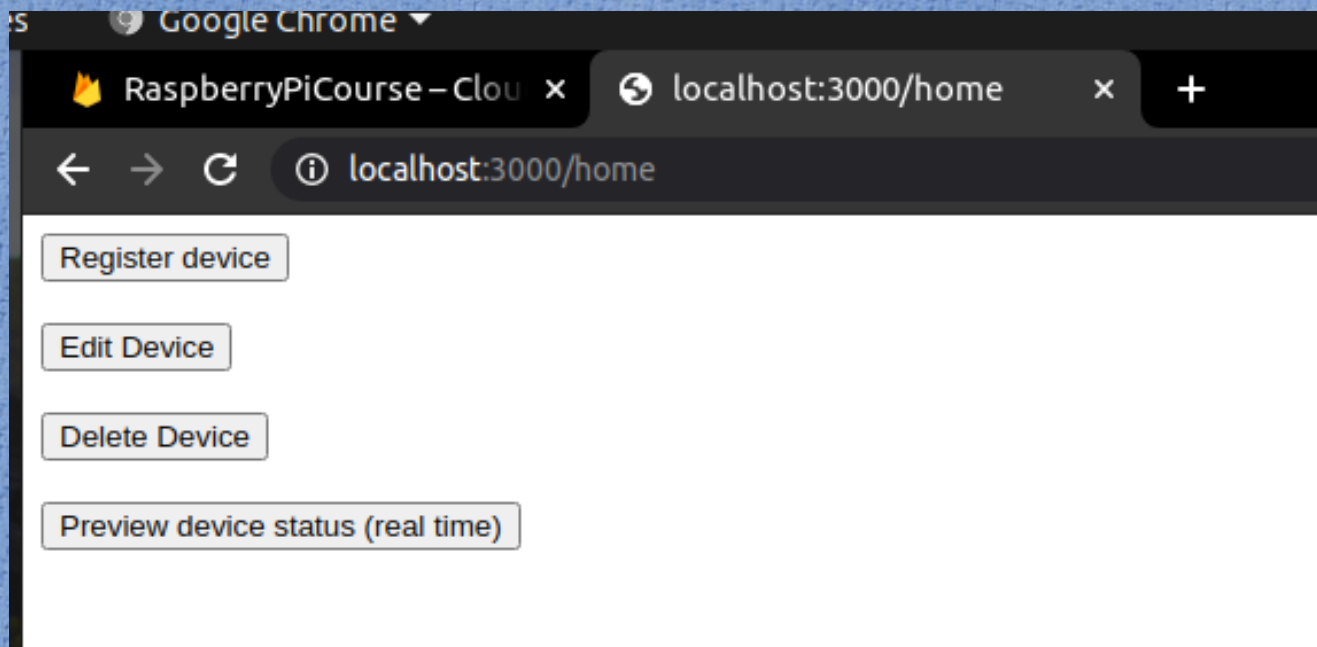
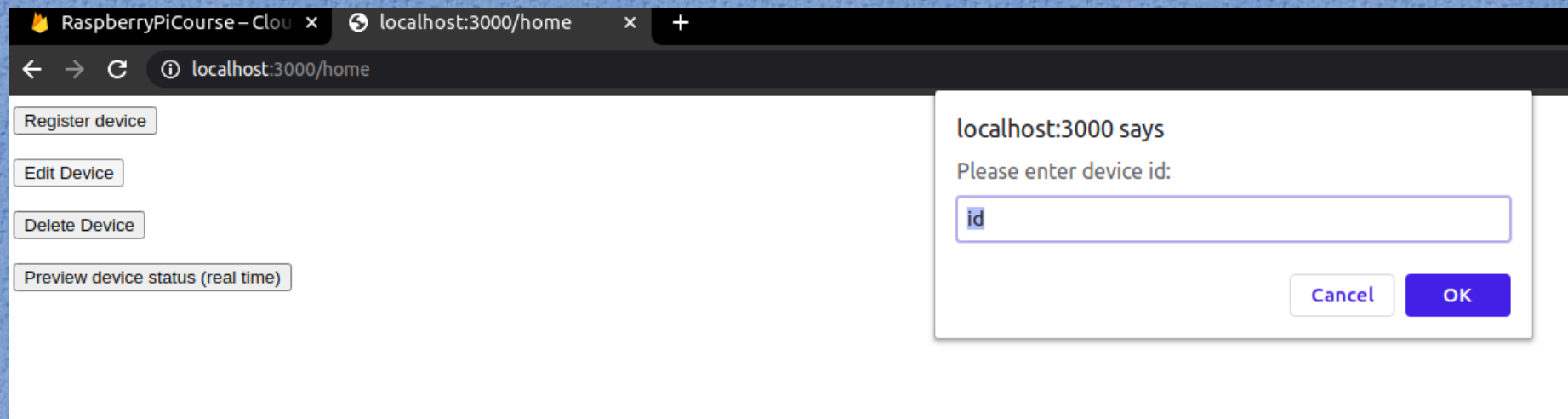
Факултет за електротехника и информациски технологии



## Задача 2

Прошири го node.js express сервер на raspberry pi од претходната задача за да може да се справи со следниве барања:

- ❑ /home – GET метод кое враќа страница со 4 копчиња. Првото копче е Register device кое служи за додавање на нови паметни уреди. Како влезни параметри потребно е да се внесат име на уред и име на корисник кој го регистрира (може да се користи prompt) и истите да се проследат на POST метод /devices. Второто копче е Preview device status (real time) и при клик на истото се повикува GET метод /devices кое префрлува на друга страница. Третото копче е Edit Device и служи за промена на податоци на веќе постоечки уред. Како влезни параметри потребно е да се внесат име, id на уред и име на корисник кој го регистрира (може да се користи prompt) и истите да се проследат на PUT метод /devices. Четвртото копче е Delete Device и служи за бришење на уред (како влезен параметар треба да се прати id на уред)
- ❑ /devices/:id – PUT метод кој ќе се користи за изменување на веќе постоечки уред (се проследува име на уред, име на корисник)
- ❑ /devices/:id – DELETE метод кој ќе се користи за бришење на уред





## Registeres Devices:

1. 1D6AikSgoDvSsgezaBd1 --> Living room light 1 edited is on
2. 3hpoFkwmFuDpbLqxPIZB --> Kitchen light 1 is on
3. eVnfFDfh0pLmr20tYT0r --> Living room light 2 is on

## History:

1. New device: Living room light 1 edited is on ( created by Tamara Mitevaska at Tue Oct 06 2020 )
2. New device: Living room light 2 is on ( created by Tamara Mitevaska at Tue Oct 06 2020 )
3. New device: Living room light 6 is on ( created by Tamara Mitevaska at Wed Oct 07 2020 )
4. New device: Living room light 8 is on ( created by Tamara Mitevaska at Wed Oct 07 2020 )
5. Modified device: Kitchen light 1 is on ( modified by Tamara Mitevaska at Wed Oct 07 2020 )
6. Removed device: Living room light 8 is on ( removed by Tamara Mitevaska at Wed Oct 07 2020 )

# *Прашања?*

**БЛАГОДАРАМ НА  
ВНИМАНИЕТО! 😊**