Echipa nr 8:Raspberry pi Traffic Lights Manager

Grupa: 1306A

Nume Studenti: Baciu C. Alexandru

Biltic Silviu-Mihai

Mail : silviu-mihai.biltic@student.tuiasi.ro

alexandru.baciu@student.tuiasi.ro

Proiect : Raspberry pi Traffic Lights

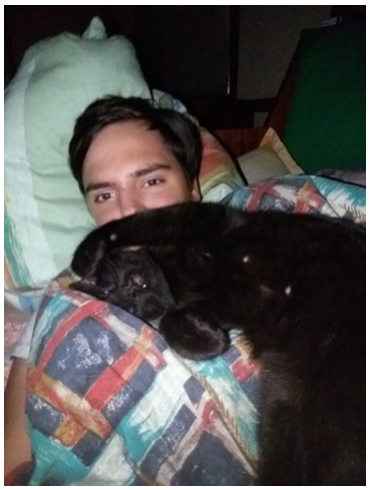
Video: https://www.youtube.com/watch?v=fAZdKKr7wsw&feature=youtu.be

Hackster: -

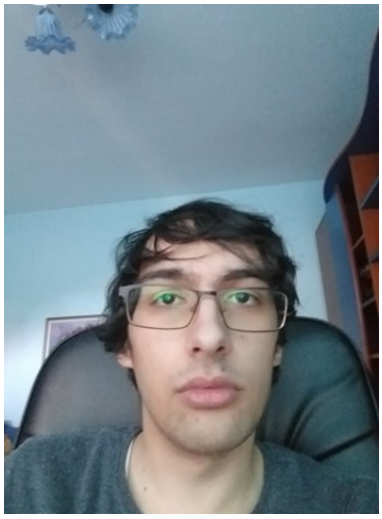
Examen Baciu C. Alexandru (nu s-a putut conecta la smlab) :https://pastebin.com/yxyMLLLQ

Poze :

Baciu C. Alexandru



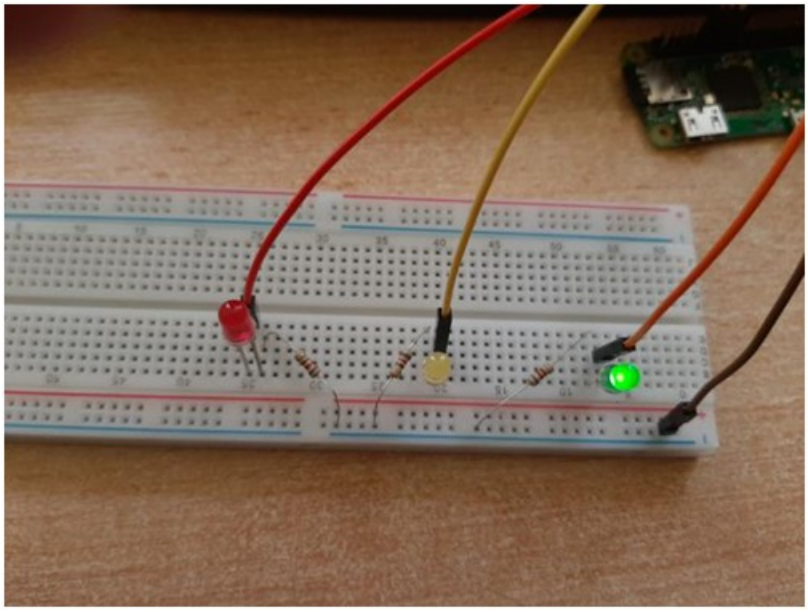
Biltic Silviu-Mihai



Elevator Pitch

Manage a traffic lights system in your browser using a RaspberryPi Zero W

Cover Image



Story

For some people it might be hard to get started with raspberry pi, especially when they don’t want to invest too much or you don’t want to start your journey with a project that’s too complicated.

This is a simple project about simulating a traffic lights system on the raspberry pi, which is controlled via a NodeJS server which is hosted on the board.

The process of using the application is simple: once you start running the server, you can connect to it using a laptop, at the address “raspberrypi:3000”. The webpage contains two buttons “ON” which starts the traffic lights and “OFF” stops them.

While the traffic lights are ON, they will change their color in order to mimic a real traffic lights behavior , looping through these states: “ Red, then red and yellow,then green, then yellow again”

Components and apps

Hardware components :

- Raspberry Pi Zero WH

- 3x Resistor 820Ω

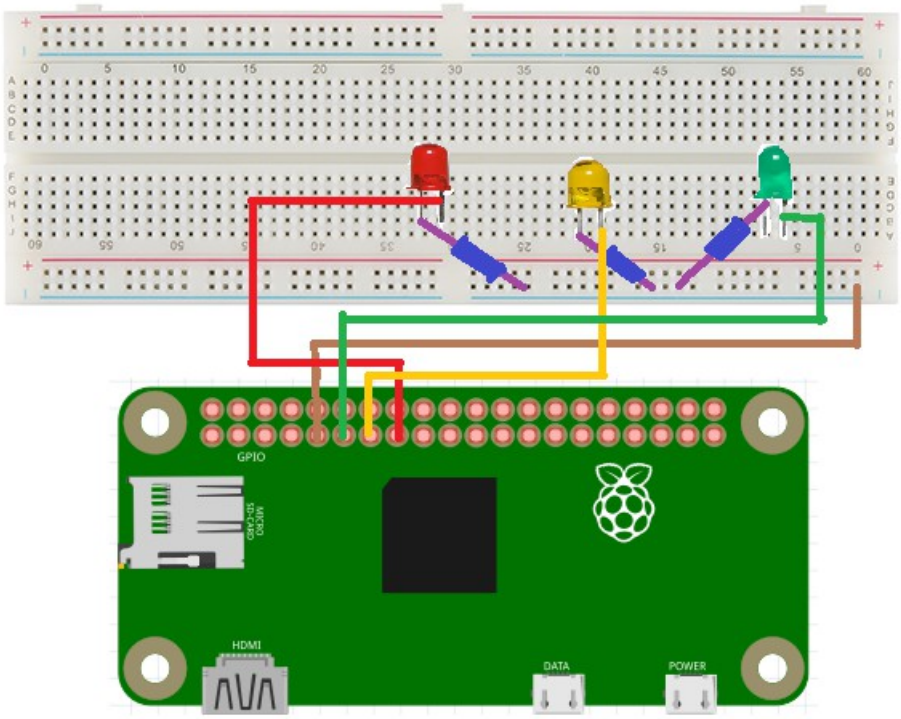
- 3x Coloured LEDs

- 4x Male/Female jumper wires

Hardware components :

-Raspbian

-Nodejs (with nodemon,ejs,session,body-parser and onoff modules).



Schematics

The LEDs are controlled by control pins 17 for green, 27 for yellow and 22 for red, connected to the cathode. The anode is connected to the table of the circuit. The table (-) of the circuit is connected to GND and the LEDs are protected using 820 ohms resistances.

How to run the program?

1. Install NodeJS (steps to do this are easily found on the web)
2. Download the code and use the command
“node [filename].js”
3. Start the program and see how it goes.

Code:

Server code:

```
var express = require('express');
var app = express();
var path = require('path');
```

```
app.set('view engine', 'ejs');
app.use(express.static(path.join(__dirname, 'public')));
console.log(path.join(__dirname, 'public'));

const gpio = require('onoff').Gpio
const rosu = new gpio(22, 'out')
const galben = new gpio(27, 'out')
const verde = new gpio(17, 'out')

//functie pentru wait
function sleep(time) {
  return new Promise((resolve) => {
    setTimeout(resolve, time)
  })
}

app.get('/', function (req, res) {
  res.render('index', { status: "Apasa Butonul" });
});

app.listen(3000, function () {
  console.log('Server Started on Port: 3000!')
})

//variabila care zice daca ruleaza semaforul
var statut = false;

app.post('/led/on', function (req, res) {
  statut = true;
  Semaforizeaza();
  return res.render('index', { status: "Semaforul Ruleaza" });
});

app.post('/led/off', function (req, res) {
  statut = false;
  offALL();
return res.render('index', { status: "Semaforul s-a Oprit" });
});
```

```
//permite o singura instanta de semafor
//sa ruleze la un moment dat ( javascript = single-threaded )
var instance = 0;

async function Semaforizeaza() {
  if (instance===0) {
    instance=1;
    console.log("Semaforul va incepe in 3 secunde");
    await sleep(3000)
    console.log("Semaforul ruleaza")
    while (statut) {
      // incepe cu rosu
      rosu.writeSync(1);
      if (statut) {
        await sleep(3000);
     } else return;

      // Rosu si galben
      if (statut) {
        galben.writeSync(1);
        await sleep(1000);
      } else return;

      // verde
      if (statut) {
        rosu.writeSync(0);
        galben.writeSync(0);
        verde.writeSync(1);
        await sleep(2000);
      } else return;

      // galben
      if (statut) {
        verde.writeSync(0);
        galben.writeSync(1);
await sleep(1000)
      } else return;
```

```
      // inchide becul galben si apoi repeta ce era mai sus
      if (statut) {
        galben.writeSync(0);
      } else return;
    }
  }else return;
}
async function offALL() {
  rosu.writeSync(0)
  galben.writeSync(0)
  verde.writeSync(0)
  instance=0;
}
//stinge luminile cand se inchide procesul (ctrl+c)
process.on('SIGINT',()=>{
 offALL();
 process.exit();
})
//La pornire asigura ca  aplicatia este in starea initiala
offALL()
```

HTML Code:

```
<meta charset="UTF-8">
<head>
  <style>
    body{
      text-align: center;
    }
    button{
      text-align: center;
      font-family: 'Franklin Gothic Medium', 'Arial Narrow', Arial, sans-serif;
      font-size: large;
      background-color: chocolate;
    }
  </style>
</head>
```

```
<body>
<div class="BorderMargin">
<image src = 'https://www.razvanorasanu.ro/wp-content/uploads/2018/03/semafor-scoal-soferi-600x60011-600x600-600x600.jpg' alt="LED" >
<h1>Bine ati venit pe serverul de semafor</h1>

<form action="/led/on" method="post">
<button type="submit" class="button"> On </button>

<button type="submit" formmethod="post" formaction="/led/off" > Off </button>
</form>
<a>Semafor Status: <%=status %></a>
</div>
</body>
```

Contributions:

Baciu C. Alexandru:
-Configurare sistem de operare Raspbian
-Asamblarea proiectului (LED-uri,rezistente,placuta)
-Codul HTML pentru pagina web
-Schematicile si videoclipul
Biltic Silviu-Mihai:
-Configurarea serverului NodeJS
-Codul pentru logica aplicatiei
-Integrarea documentatiei