

---

**LeDi**

***Release 0.1***

**LeDi Team**

**Dec 18, 2022**



# CONTENTS

<b>1</b>	<b>LeDi.Docs</b>	<b>3</b>
<b>2</b>	<b>LeDi.Server</b>	<b>5</b>
<b>3</b>	<b>LeDi.WebClient</b>	<b>7</b>
<b>4</b>	<b>LeDi.Display</b>	<b>9</b>
<b>5</b>	<b>LeDi.Hardware</b>	<b>11</b>
5.1	Index . . . . .	11



**LeDi** is an open source Soft- and Hardware project to primary manage sport matches. It is supported by a LED display to show the current time left, player information, scores etc..

Project at GitHub: [LeDi](#)

The project has multiple subprojects:



**LEDI.DOCS**

This projects are the documentation like this. It documents how to setup the software and how to build the hardware. It documents the road of custructing the hardware until the final result came out.





## **LEDI.SERVER**

This is the main software component. It is the core of the software architecture and provides an API to provide the current match status. The API can be accessed via a REST API. LeDi.Server is the only software component that holds the valid match information. It is created using C# based on .Net 6.0 (LTS).



## **LEDI.WEBCLIENT**

This is the software component that provides the frontend to the referees and all other people. The referees can control the match using this component while other people can watch the current standings. Admins configure the software after installation using the LeDi.WebClient. It is created in C#, HTML, CSS and Java Script based on Microsoft Balzor (Server) and .Net 6.0 (LTS).



## **LEDI.DISPLAY**

This software component controls the LED display, based on WS2812B LEDs. It connects to the LeDi.Server API and gets the information about the behaviour and what needs to be shown from this server also via REST API. It is created using C# based on .Net 6.0 (LTS).



## LEDI.HARDWARE

This project documents the creation of the display hardware. This is the hardware counterpart to the LeDi.Display project. It contains the mounting of the case and power supplies as well as the drafts for the electronics and cableing. Especially electronics and calbing should be supported by a trained electican to exclude damage to your health due to electic shocks or other dangers during build and run.

---

**Note:** This project and all subprojects are currently under developement. For a roadmap see the roadmap in GitHub.

---

### 5.1 Index

#### 5.1.1 Idea

Todo

#### 5.1.2 LeDi.Server

##### Installation

To use Lumache, first install it using pip:

```
(.venv) $ pip install lumache
```

##### Creating recipes

To retrieve a list of random ingredients, you can use the `lumache.get_random_ingredients()` function:

The `kind` parameter should be either `"meat"`, `"fish"`, or `"veggies"`. Otherwise, `lumache.get_random_ingredients()` will raise an exception.

For example:

```
>>> import lumache
>>> lumache.get_random_ingredients()
['shells', 'gorgonzola', 'parsley']
```

### 5.1.3 LeDi.Display

### 5.1.4 LeDi.Hardware

### 5.1.5 LeDi.Docs

### 5.1.6 Install LeDi on a RaspberryPi

This guide will lead you through the process of installing LeDi on a RaspberryPi from scratch. This guide was created using a RaspberryPi 3 but this should also be valid for a RaspberryPi 4. For RaspberryPi 1 and 2 you might need an alternative solution for hosting the Wifi Hotspot by adding a USB wifi stick or by adding an access point to the network. As we only recommend to use a RaspberryPi 3 or 4, we will not go into details how to implement this for RaspberryPi 1 or 2.

#### Install the RaspberryPi

This section needs to be improved to go into more details (i.e. hostname, IP, configure User, Wifi and SSH on image creation). To install the RaspberryPi, follow the regular process of installing it. Use a 32 Bit “Lite” image. The lite image does not have any desktop environment installed, which saves resources because it is not needed. Also run the following commands after installing OS: .. code-block:: bash

```
linenos
```

```
sudo apt update sudo apt upgrade
```

#### Install .Net 6:

We will install .Net 6 now. To start, go to <https://dotnet.microsoft.com/en-us/download/dotnet/6.0>. On that page select at “Linux” OS the Binary “Arm32”. Copy the URL, that is the “direct link” (i.e. <https://download.visualstudio.microsoft.com/download/pr/5a24144e-0d7d-4cc9-b9d8-b4d32d6bb084/e882181e475e3c66f48a22fbfc7b19c0/dotnet-sdk-6.0.400-linux-arm.tar.gz>). Now connect to your RaspberryPi and run .. code-block:: bash

```
linenos
```

```
wget <url from above>
```

Now run the following commands. Replace the filename at the first line with the filename you just downloaded before.  
.. code-block:: bash

```
linenos
```

```
DOTNET_FILE=dotnet-sdk-6.0.100-linux-x64.tar.gz export DOTNET_ROOT=$(pwd)/.dotnet  
mkdir -p "$DOTNET_ROOT" && tar xzf "$DOTNET_FILE" -C "$DOTNET_ROOT"  
export PATH=$PATH:$DOTNET_ROOT:$DOTNET_ROOT/tools
```

Congratulations. Dotnet is installed.



## Setup the Access Point

This section will setup the RaspberryPi as a stand alone Hotspot. You can skip this, if you don't plan to use LeDi stand alone. Otherwise if you plan to just take your display and go to the next court, pool, field or whatever - you should implement this. To connect to the Raspberry via wifi where the RaspberryPi acts as the access point, we need to perform the following steps. To avoid connection loss, connect the RaspberryPi via Ethernet cable to your network or PC. Summarized we will setup the RaspberryPi as a DNS server (dnsmasq) and DHCP server (dhcpd) and install hostapd to use the wifi adapter to provide a wifi to clients to connect to it.

### Install Hostapd and Dnsmasq:

Run the following commands to install hostapd and dnsmasq and configure hostapd. dhcpd is already installed by default. .. code-block:: bash

**linenos**

```
sudo apt install hostapd -y sudo apt install dnsmasq -y sudo systemctl unmask hostapd sudo systemctl enable hostapd
```

After the installation is done, stop the services, so we can edit the config files after: .. code-block:: bash

**linenos**

```
sudo systemctl stop hostapd sudo systemctl stop dnsmasq
```

### Configure DHCP (dhcpd)

Now we setup the private network for the Wifi interface of the Raspberry Pi. We will choose a subnet in one of the private IP segments and a /24 (255.255.255.0) subnetmask. We will take 10.10.100.1 for the IP address of the Raspberry Pi Wifi interface. Feel free to choose something else. Edit the IP configuration as follows: .. code-block:: bash

**linenos**

```
sudo nano /etc/dhcpd.conf
```

Add the following lines at the end of the file: .. code-block:: text

**linenos**

```
interface wlan0 nohook wpa_supplicant static ip_address=10.10.100.1/24 static domain_name_servers=10.10.100.1
```

Save the file with Ctrl+O and exit with Ctrl+X

### Configure DHCP (dnsmasq)

Now we configure the DHCP server which will serve the IPs to the connecting clients. Create a new config first. .. code-block:: bash

**linenos**

```
sudo mv /etc/dnsmasq.conf /etc/dnsmasq.conf.orig sudo nano /etc/dnsmasq.conf
```

Now paste the following content: .. code-block:: text

**linenos**

```
interface=wlan0 dhcp-range=10.10.100.11,10.10.100.254,255.255.255.0,1h domain=intern listen-address=10.10.100.1 listen-address=127.0.0.1 local=/intern/
```

Save the file with Ctrl+O and exit with Ctrl+X. That lines mean that we will serve IPs from 10.10.100.11 to 10.10.100.254 in the subnet 255.255.255.0 with 1 hour of lease time to connecting clients. We leave the first 10 IPs free for infrastructure devices we may connect somewhen. It also defines, that also other systems than the local system can connect to the dnsmasq-service to get DNS information as well as that the “intern”-Domain should not be resolved by anyone else than the local system (in case it will have an uplink while it runs).

### Configure DNS (dnsmasq)

Now we configure the DNS resolution. Edit the hosts file: .. code-block:: bash

```
linenos
```

```
sudo nano /etc/hosts
```

Add the lines at the end: .. code-block:: text

```
linenos
```

```
10.10.100.1 LeDi.intern 10.10.100.1 board board.intern
```

The last line would mean, that the system will have an alias called “board” that clients are able to use to connect to the page. Feel free to change it or add more/other alias’. Save the file with Ctrl+O and exit with Ctrl+X.

### Configure Access Point (hostap)

After that we configure the AccessPoint itself. .. code-block:: bash

```
linenos
```

```
sudo nano /etc/hostapd/hostapd.conf
```

Paste the following lines: .. code-block:: text

```
linenos
```

```
interface=wlan0 hw_mode=g channel=7 wmm_enabled=0 macaddr_acl=0 auth_algs=1 ig-  
nore_broadcast_ssid=0 wpa=0 ssid=LeDi
```

Save the file with Ctrl+O and exit with Ctrl+X. This will configure the interface wlan0. The connection is named “LeDi” by default and unencrypted so everyone will be able to connect.

Link the new config from the initscript file: .. code-block:: bash

```
linenos
```

```
sudo nano /etc/default/hostapd
```

Replace the Line **#DAEMON\_CONF=""** with **DAEMON\_CONF="/etc/hostapd/hostapd.conf"** to remove the leading # and add the path the new config file. Save the file with Ctrl+O and exit with Ctrl+X.

## Finalize

(Re-)Start the Services: .. code-block:: bash

**linenos**

```
sudo systemctl restart dhcpd sudo systemctl start hostapd sudo systemctl start dnsmasq
```