

GEVORDERDE WEBAPPLICATIES

Verslag Project

Tarik Kaya,
10 November 2022

Inhoudstafel

1. INLEIDING	3
2. DATACOMMUNICATIESTANDAARDEN	3
3. COAP	4
4. FLASK.....	4
4.1 API	4
4.2 LOGIN.....	5
4.3 BCRIPT	5
5. ASYNCIO EN SOCKETIO.....	5
5.1 ASYNCIO.....	5
5.2 SOCKETIO	6
5.3 ASYNCIO EN SOCKETIO	6
6. JSON	6
7. FRONT-END DEVELOPMENT	7
7.1 HTML EN CSS	7
7.2 JQUERY	7
7.3 AJAX	7
7.4 PROJECT LAYOUT	7
8. DATABASE MANAGEMENT.....	10
9. DOCKER.....	10
BRONVERMELDING	12

1. Inleiding

In dit project is het de bedoeling om een webapplicatie te maken waarbij lampen kunnen aangestuurd worden gebruik makende van COAP. Om de lampen aan te sturen en te controleren wordt een api gemaakt via een restful manier. Lampen worden aangesproken via get, put en post commands.

2. Datacommunicatiestandaarden

Onderstaande figuur toont voor elke laag van het OSI-model protocollen die geschikt zijn voor low power WPAN:

Application	Websocket, HTTP, COAP
Transport	UDP, TCP
Network	RPL
Adaptation	6LoWPAN
MAC	CSMA
Radio Duty Cycling	NullRDC, ContikiMAC, X-MAC
Physical	IEEE 802.15.4

Figuur 1: protocollen voor WPAN, per laag van het OSI-model (1)

Merk op dat dit niet het standaard OSI-model is. De Session en de presentation layer zijn ingekapseld in de application layer. Bovendien is er ook een adaptation layer voor fragmentatie van de inkomende data. Elke hogere laag vraagt een service aan de lagere laag. De lagere laag opereert als een black box, laag n+1 is dus niet op de hoogte van wat er gebeurt op laag n. Het is in feite alsof bij elke laag de datatransfer horizontaal gebeurt, maar de transfer gebeurt verticaal tot de fysieke laag, en kan dan via de fysieke laag getransmitted worden tot de receiver.

Zo vraagt COAP bijvoorbeeld aan de transportlaag om de data te transporteren. COAP voegt zijn header toe aan de data en geeft de data door aan de transportlaag. Het transport gebeurt connection-oriented (TCP) of connectionless (UDP). De transportlaag

overhandigt de routing aan de networklaag. Dit is een connectionless protocol en het zou dus perfect kunnen gebeuren dat een pakket verloren geraakt. Tenslotte wordt CSMA (of TSCH) gebruikt als MAC-protocol. De data wordt dan overhandigt aan de fysische laag, om doorgestuurd te worden naar de receiver. Bij de receiver wordt de data layer per layer terug naar boven gehaald. Elke laag gebruikt de bijhorende headers om de informatie te extraheren. Voor de fysische laag wordt IEEE 802.15.4 (low-power WPAN) gebruikt.

3. COAP

COAP staat voor Constrained Application Protocol. Het is een client-server protocol. Bij COAP kan een client node een andere client de opdracht geven om een COAP-pakket te verzenden. De COAP-server zal het pakket dan interpreteren, de data extraheren en beslissen wat doen.

COAP kan gezien worden als de light weight versie van HTTP. COAP opereert op resources. Op deze resources kunnen methoden toegepast worden. In het totaal zijn er 4 methoden:

- `get()`: vraagt informatie op
- `put()`: nieuwe waarde toekennen aan een resource
- `post()`: nieuwe resource aanmaken en een waarde aan toekennen
- `delete()`: verwijdert de resource

COAP heeft een identifier: URI. Deze is vergelijkbaar met URL bij HTTP.

Via COAP is het ook mogelijk om te multicasten. Zo kan je bijvoorbeeld methoden toepassen op een volledige rij LEDs, zoals het verlichtingsniveau aanpassen of de LEDs volledig uitschakelen.

COAP gebruikt als transportprotocol UDP en is dus connectionless. Per definitie is COAP dus ook stateless, net zoals HTTP.

4. Flask

4.1 API

Flask is een framework om API's aan te maken in python. Het is zeer uitgebreid en ook eenvoudig te gebruiken. Zo kan je op eenvoudige manier URL routes aanmaken en refereren naar de juiste html pagina. Je kan ook methodes zoals `get`, `put` en `post` gebruiken om op een restful manier te gaan werken. Rest staat voor Representational state transfer. REST is een gestandaardiseerd architectuur die resources makkelijk kan identificeren en bewerken.

4.2 Login

Flask-login biedt een algemeen framework voor het in- en uitloggen van gebruikers. Indien gebruikers niet ingelogd zijn, is het ook mogelijk om toegang tot bepaalde routes van de flask-api te blokkeren. Flask kan de sessie onthouden zodat de gebruiker gedurende lange perioden ingelogd kan blijven.

4.3 Bcrypt

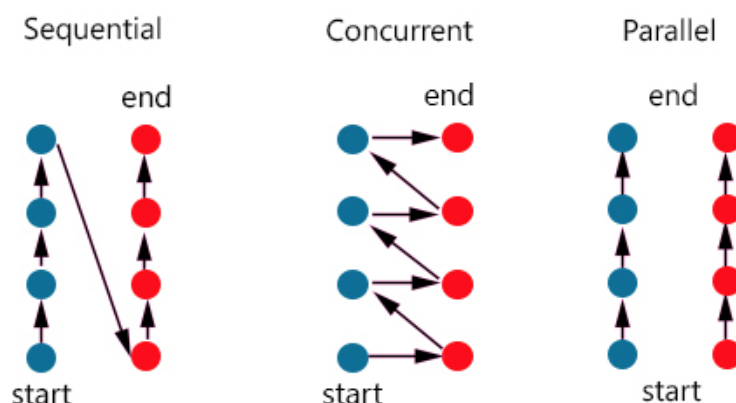
Flask bcrypt maakt gebruik van een hashfunctie om een wachtwoord te hashen. Cryptografische hashfuncties zorgen ervoor dat de integriteit van de data wordt bewaard. De integriteit van data is het verifiëren dat de data niet veranderd of vernietigd is. Het kraken van originele informatie is dus zeer moeilijk, zelfs met moderne grafische kaarten aangezien bcrypt is specifiek gedesigned om traag te werken.

Een probleem bij het onmiddellijk hashen van het wachtwoord is dat indien 2 gebruikers dezelfde wachtwoord hebben, de hash dan ook gelijk zal zijn. Om dit te vermijden wordt salting gebruikt. Bij elk wachtwoord zal een unieke string van minstens 32 karakters toegevoegd worden aan het wachtwoord en het geheel zal dan gehashed worden. Op deze manier hebben niet alleen 2 wachtwoorden een andere hash, maar is brute force en reverse engineering ook veel moeilijker.

5. Asyncio en Socketio

5.1 Asyncio

Asyncio is een python library dat wordt gebruikt voor het asynchroon programmeren. Asynchroon betekent dat er gelijktijdige uitvoeringen zijn die in een willekeurige volgorde ten opzichte van elkaar kunnen worden uitgevoerd. Bij asynchrone functies wordt de functieaanroep gedaan, maar wordt er niet gewacht tot de aanroep is voltooid. De status of het resultaat van de aanroep kan later opgevraagd worden. In het engels spreekt men over "concurrency".



Figuur 2: Concurrency en parallelism (2)

Bij asyncio spreekt men typisch over coroutines, of asynchrone subroutines. Coroutines kunnen op verschillende plaatsen ingevoerd, verlaten of hervat worden. Een coroutines kan op 2 manieren uitgevoerd worden:

- `asyncio.run()`: Typisch is dit een top-level call naar een asynchrone functie. Het blokkeert de code totdat alle coroutines zijn voltooid.
- `await`: Binnen een coroutine kan je `await` gebruiken om de huidige functie te stoppen, en `asyncio` zal de functie op een later tijdstip hervatten. Dit gebeurt binnen de `asyncio.run()` functie

In ons project wordt `asyncio` gebruikt om COAP requests te versturen. Dit moet asynchroon aangezien er meerdere requests tegelijk kunnen worden opgestuurd. Bovendien moeten ook andere mensen tot de lampen toegang kunnen krijgen. Dit kan aangezien COAP volledig connectionless is en dus geen TCP-verbinding vereist. Het kan dus best zijn dat de request verloren gaat en wijzigingen aan de lampen niet worden doorgevoerd.

5.2 Socketio

Socketio is een library dat bidirectionele communicatie tussen client en server mogelijk maakt. Bidirectionele communicatie is mogelijk wanneer een client socketio in de browser heeft en de server socketio heeft geïntegreerd. De communicatie is volledig event-based. Data kan in meerdere standaarden doorgestuurd worden. JSON is de eenvoudigste formaat.

Socketio kan ook gebruikt worden voor het beheren van websockets, maar niet alleen dat. In tegenstelling tot websockets kan Socketio een bericht uitzenden naar alle aangesloten clients. Indien in de webapplicatie een client een bepaalde lamp zou aanpassen, wordt de verandering gebroadcast naar alle andere clients.

Als bij een websocket de verbinding wegvalt, zal de verbinding niet automatisch herstart worden. Bij Socketio is dat wel het geval.

5.3 Asyncio en Socketio

Asyncio en socketio zijn libraries die conflicten genereren wanneer ze tegelijk worden gebruikt. De `asyncio await` functie werkt binnen flask maar niet met Socketio. Om dit probleem op te lossen moet `asyncio.run()` gebruikt worden.

6. Json

Json staat voor Javascript Object notation. Het is een tekst gebaseerde manier om data binnen Javascript door te sturen. Het is een lightweight formaat om tijdelijke data te beheren. JSON zet complexe data om in een begrijpelijk en leesbaar formaat. Data is opgeslaan als key-value paren.

Een andere alternatief om data op te slaan is XML. XML staat voor extensible markup language. XML is vooral gemaakt om gegevens te vervoeren, en niet om gegevens weer te geven. Het is een opmaaktaal die een standaard definieert voor het encoderen van data.

7. Front-end development

7.1 HTML en css

HTML staat voor Hyper Tekst Markup Language en wordt gebruikt voor het opmaken van webpagina's en webapplicaties. Het bepaalt de structuur van webpagina's. Om een bepaalde layout te geven aan een webpagina wordt css gebruikt.

HTML is enkel een statische template. Om dynamische elementen toe te voegen wordt gebruikt gemaakt van Javascript.

7.2 JQuery

Jquery is een Javascript bibliotheek dat ontworpen is om de client-side scripting van HTML te vereenvoudigen. Het maakt het manipuleren van event-handling en animaties zeer gemakkelijk. In dit project wordt het voornamelijk gebruikt om de waarden van lampen en sliders visueel aan te passen.

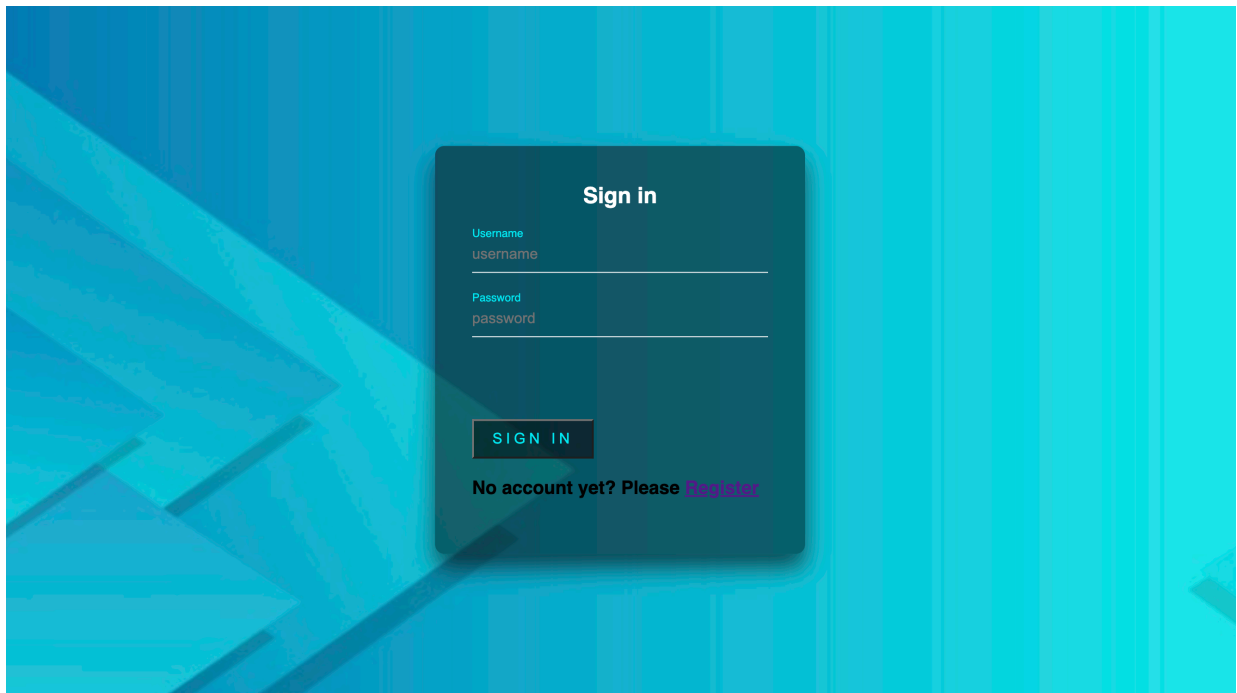
7.3 Ajax

Ajax staat voor Asynchronous Javascript en XML. Ajax stuurt asynchrone http-request om dan als antwoord data in XML-formaat terug te krijgen. De nodige onderdelen van de webpagina kunnen dan aangepast worden, zonder dat de pagina opnieuw moet herladen worden.

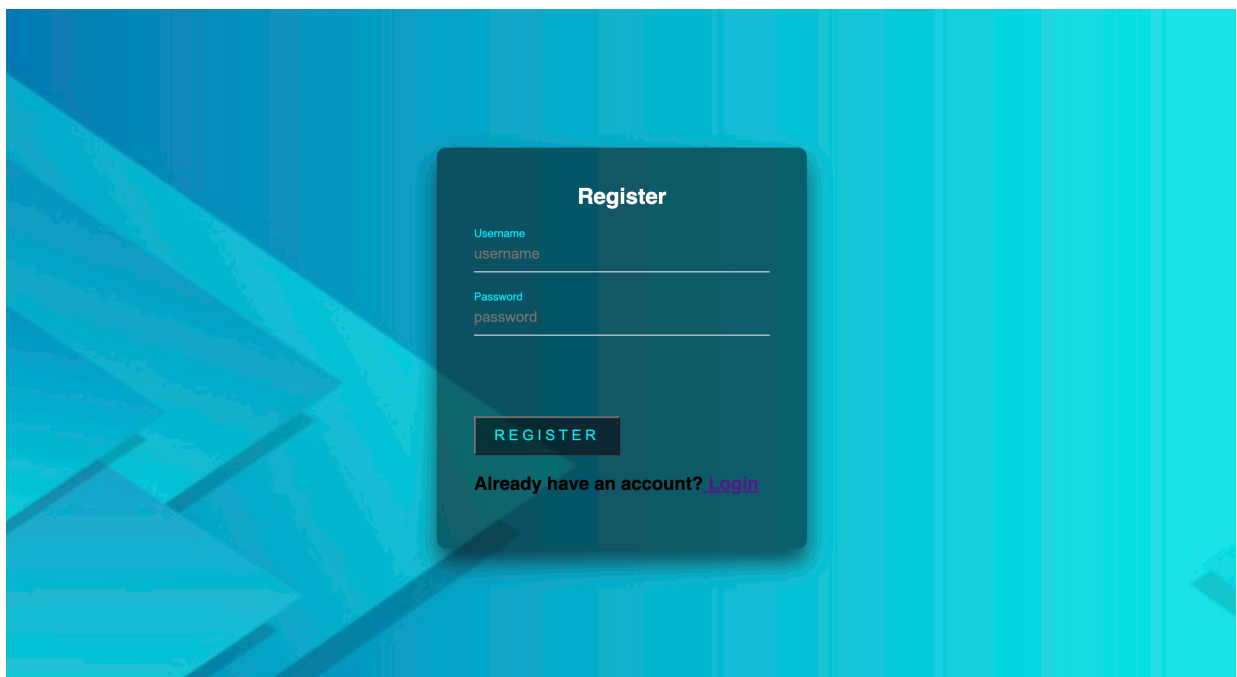
Ajax wordt gebruikt om de status van de lampen aan te vragen en aan te passen. Hiervoor worden get en post request gestuurd naar de api van de webserver. Voor elke specifieke lamp kan een specifieke URL meegegeven worden.

7.4 Project Layout

De webapplicatie bestaat uit een register pagina, login pagina, een dashboard voor de lampen en een history pagina waarbij alle logs van de lampen staan. De design werd gemaakt via HTML en css, met behulp van Bootstrap dat een series van tools bevat om de design te vergemakkelijken.



Figuur 3: Login pagina



Figuur 4: Register pagina



Figuur 5: Dashboard

Lamp change history

DATE	USERNAME	LAMP	VALUE
11/13/2022, 19:04:00	anne	All lamps	84
11/13/2022, 19:03:58	anne	lamp1b	54
11/13/2022, 18:48:15	anne	All lamps	76
11/13/2022, 18:48:12	anne	lamp1c	38
11/13/2022, 18:47:42	anne	lamp1b	52
11/13/2022, 18:45:31	test4	lamp1b	86
11/13/2022, 17:26:58	arda	lamp2b	82
11/13/2022, 17:24:02	tarik	All lamps	84
11/13/2022, 17:23:55	tarik	lamp1c	55
11/13/2022, 16:59:21	tarik	All lamps	90
11/13/2022, 16:59:13	tarik	lamp1c	86
11/13/2022, 16:59:05	tarik	lamp1b	58
11/13/2022, 16:51:39	sara	All lamps	29
11/13/2022, 16:51:36	sara	lamp1b	75

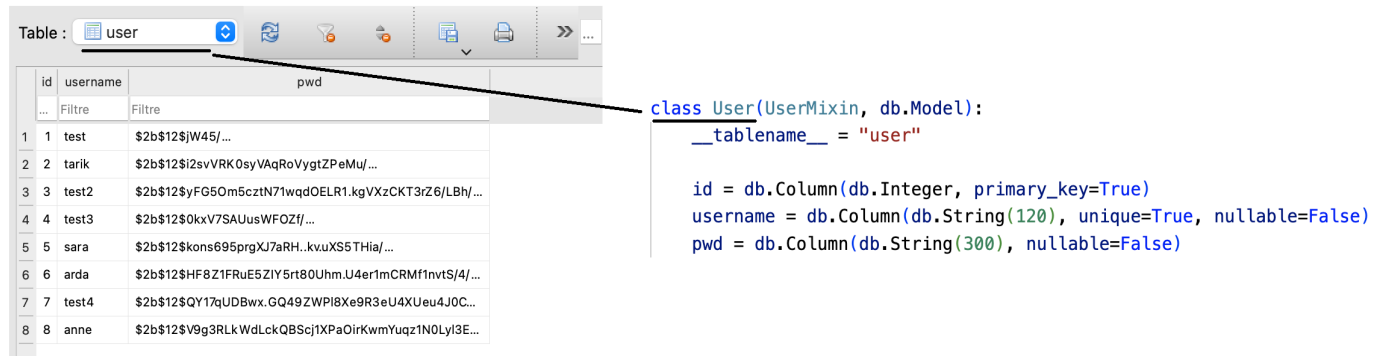
[HOME](#) [LOGOUT](#)

Figuur 6: history logs lampen

8. Database management

Er werd nooit expliciet een database aangemaakt, noch expliciet SQL commando's gebruikt. Om een database model te bouwen werd gebruik gemaakt van ORM (Object Relational Mapping). ORM biedt een brug tussen een relationele database en python-objecten. Het opvragen van de data uit ORM is ook veel eenvoudiger dan SQL commands.

Volgende figuur toont de mapping tussen de relationele database en de python class:



The figure shows a mapping between a database table and a Python class. On the left, a table named 'user' is displayed with columns 'id', 'username', and 'pwd'. On the right, the Python class 'User' is defined, inheriting from 'UserMixin' and 'db.Model'. The class has attributes 'id', 'username', and 'pwd' defined as database columns.

	id	username	pwd
1	1	test	\$2b\$12\$JW45/...
2	2	tarik	\$2b\$12\$i2svVRK0syVAqRoVygZPeMu/...
3	3	test2	\$2b\$12\$yFG5Om5cztN71wqdOELR1.kgVxzCKT3rZ6/LBh/...
4	4	test3	\$2b\$12\$0kxV7SAUusWFOZf/...
5	5	sara	\$2b\$12\$kons695prgXJ7aRH..kvuXS5THia/...
6	6	arda	\$2b\$12\$HF8Z1FRuE5ZIY5rt80Uhm.U4er1mCRMf1nvtS/4/...
7	7	test4	\$2b\$12\$QY17qUDBwx.GQ49ZWPI8Xe9R3eU4XUeu4J0C...
8	8	anne	\$2b\$12\$V9g3RLkWdLckQBScj1XPaOirKwmYuqz1N0Lyl3E...

```
class User(UserMixin, db.Model):  
    __tablename__ = "user"  
  
    id = db.Column(db.Integer, primary_key=True)  
    username = db.Column(db.String(120), unique=True, nullable=False)  
    pwd = db.Column(db.String(300), nullable=False)
```

Figuur 7: ORM

In dit voorbeeld erft de klasse User van Usermixin, dat standaardimplementaties biedt voor alle eigenschappen en methoden van flask-login.

9. Docker

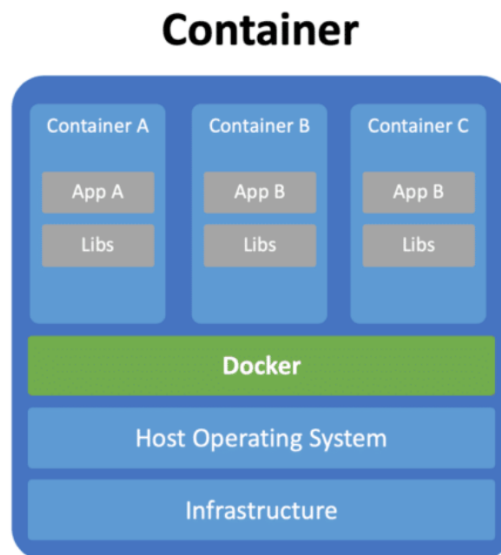
De webapplicatie bevat heel veel libraries zoals flask, asyncio, ...

Deze mogen geen conflicten genereren en hebben soms specifieke versies nodig. Hierbij hoort vaak ook een specifieke python versie. Bijkomend kunnen er ook nog externe programma's zoals RabbitMQ aanwezig zijn. Om dit op te lossen worden containers gebruikt.

Docker voegt een laag toe tussen de computer en de app, dat een virtualisatie aanbiedt op het niveau van de operating system. Eerst moet een docker image aangemaakt worden met de bijhorende requirement txt file en de te runnen python file. Zodra de container image is aangemaakt, kan deze uitgevoerd worden.

```
1 FROM python:3.8
2 ADD tarik_kaya_gvwa/requirements.txt ./webapp/requirements.txt
3 RUN pip install -r ./webapp/requirements.txt
4 ADD tarik_kaya_gvwa/ ./
5 CMD python3 webserver3.py
6 EXPOSE 8081/TCP
```

Figuur 8: Dockerfile



Figuur 9: Docker container

Bronvermelding

- (1) Toepassing van het OSI-model op IOT networks (geraadpleegd op 11/11/2022)
<https://jwcn-urasipjournals.springeropen.com/articles/10.1186/s13638-020-1645-4>
- (2) Verschil tussen parallelism en concurrency (geraadpleegd op 11/11/2022)
<https://dsin.wordpress.com/2017/08/28/different-between-concurrency-and-parallelism/>