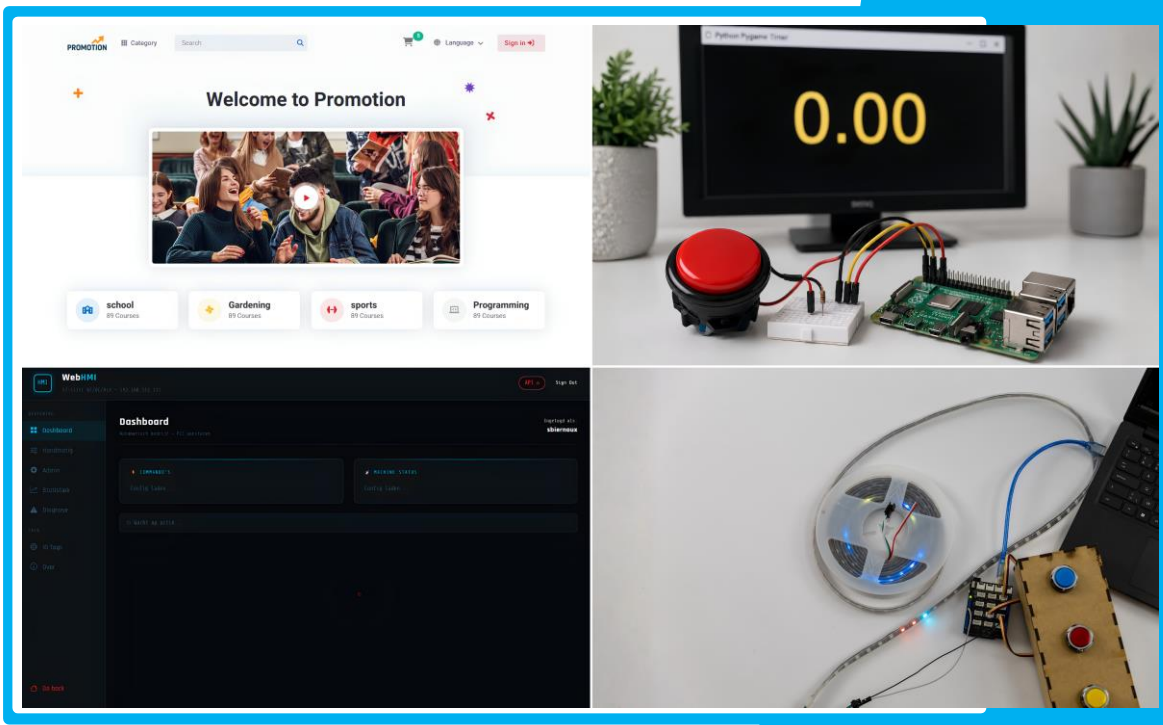
GIP BUNDEL

Biernaux Sem



2025-2026

Geïntegreerde proef

Studierichting: *6ICW*

Begeleiders:

C. Gratessolle & F. Meyers

PROVIL
UNIEK IN STEM & SPORT!

Voorwoord

Beste lezer

Wanneer u verder leest, zal u alles te weten komen over wat ik dit jaar heb mogen maken als geïntegreerde proef. Ook krijg ik de kans om alles te laten zien wat ik heb bijgeleerd in de richting informatica en communicatiewetenschappen . Dit jaar hebben wij in plaats van één groot project er vier gemaakt. De projecten waren ieders uniek en overlapten elkaar soms in manier van opbouw. Ook waren alle projecten een nieuwe uitdaging, wat ik wel leuk vond. Natuurlijk waren sommige projecten leuker dan andere, maar ik ben toch heel blij dat ik ze alle vier heb mogen maken.

Graag wil ik iedereen bedanken die mij doorheen dit traject heeft begeleid.

Als eerste wil ik meneer Christophe Gratessolle bedanken, leerkracht web- en app-development en leerkracht ICT-projecten + STEM. Meneer Gratessolle heeft ons veel bijgeleerd over het bouwen van websites tot alles in databases.

Ook wil ik meneer Frank Meyers bedanken, leerkracht programmeerwetenschappen, coderen met Python. Het opvallende aan meneer Meyers' lessen was dat deze lessen opvallend snel voorbijgingen. Ook heb ik hier geleerd hoe je gestructureerd codeert met Python.

Meneer Koen Cochet, leerkracht elektriciteit en leerkracht PC- en netwerkwetenschappen, door uw lessen heb ik netwerken kunnen opstellen en veel bijgeleerd over hoe alles in zijn werking gaat. Ook het elektriciteitsaspect heb ik van u geleerd, wat mij indirect heeft geholpen bij het maken van deze projecten.

Meneer Koen Boeckx, desondanks ik u alleen had voor elektronica, wil ik u toch bedanken voor het ondersteunen van mijn keuzes doorheen het jaar en dat ik altijd in uw lessen mocht binnenvallen!

Bedankt aan alle taalleerkrachten, mevrouw An Vos, Leen Verwimp en Martine Janssens. Jullie hebben mij en alle andere klasgenoten geholpen bij het schrijven van onze gip-bundels . Door jullie heb ik zeker veel bijgeleerd!

Als afsluiter wil ik iedereen bedanken die mij heeft geholpen bij dit traject. Van back-end tot front-end, van software tot hardware. Het was met momenten een hele uitdaging, maar het is gelukt, mede dankzij jullie.

Bedankt voor jullie hulp en veel leesplezier.

Met vriendelijke groeten
Sem Biernaux, 6ICW

Abstract

Sem Biernaux, Computer Science & Communication Sciences, Provinciaal Instituut Lommel
Abstract of research paper, submitted 24th of June 2026.

In this research paper, four projects will be discussed that differ in many ways. These projects focus on learning and applying skills related to the fields of computer science and communication sciences. The first project, called *Promotion*, focuses on using PHP and MySQL to understand databases and combine the front-end with the back-end. The second and third projects, developed using Arduino and Raspberry Pi, focus on hardware, microcontrollers, and programming languages such as Python and C++. The *WebHMI*, a collaboration with Vincent Smetsers, a student in mechatronics, combines all of these practices. The *WebHMI* controls a plc and allows it to be operated remotely from anywhere in the world.
The use of stylesheet languages such as CSS and frameworks like Bootstrap are important for optimising and neatly styling websites, which were important aspects in all four projects.

Inhoud

Inleiding

Voor mijn geïntegreerde proef heb ik dit jaar niet één groot project gemaakt, maar vier afzonderlijke projecten. Elk project bracht een eigen uitdaging mee en liet me andere vaardigheden uit de richting informatica en communicatiewetenschappen toepassen. In deze bundel beschrijf ik per project wat de opdracht was, hoe ik te werk ben gegaan en welk resultaat ik heb behaald.

Het doel van deze bundel is om aan te tonen wat ik dit jaar heb bijgeleerd, zowel op het vlak van software als hardware. Ik wil laten zien hoe ik de theorie uit de lessen heb omgezet in werkende projecten, en hoe de verschillende projecten elkaar soms aanvulden in hun opbouw. Daarnaast probeer ik telkens eerlijk te beschrijven wat goed liep en wat beter kon, zodat duidelijk wordt hoe ik tijdens het maken ben bijgestuurd.

De bundel is opgebouwd uit vier hoofdstukken, één per project. Het eerste project, *Promotion*, is een website die ik met PHP en MySQL heb gemaakt om de front-end en de back-end samen te brengen en databases te leren beheren. Het tweede project is een *reactiespel* op basis van een Raspberry Pi, waarbij hardware en programmeren samenkomen. Het derde project, de *Web HMI*, is een samenwerking met Vincent Smetsers uit de richting mechatronica. Hierbij stuur ik een plc aan die van overal ter wereld op afstand bediend kan worden, en komen zo goed als alle eerdere vaardigheden samen. Het vierde project is een spel op een *led strip*, gemaakt met een Arduino UNO en uitgewerkt in Python en C++.

Ik heb me in deze bundel beperkt tot de essentie van elk project. De volledige code en enkele bijkomende details zijn terug te vinden in de bijlagen. Op die manier blijft de tekst overzichtelijk en kan de lezer per project gericht meer informatie opzoeken.

1 Project - Promotion

1.1 Inleiding

1.1.1 Projectcontext

De richting 6SPB heeft een website nodig om hun GIP op een aantrekkelijke en overzichtelijke manier voor te stellen. De leerlingen van SPB ontwikkelen zelf verschillende cursussen die gevolgd kunnen worden. Deze cursussen bevatten oefeningen waarmee gebruikers specifieke spiergroepen kunnen trainen.

1.1.2 Doelstelling van het project

Figuur 1

Het doel van ons, 6ICW, is om een toegankelijke en gebruiksvriendelijke website te ontwerpen en te ontwikkelen, zodat 6SPB hun GIP eenvoudig kan presenteren aan een breder publiek. Voor de technische uitwerking maken we gebruik van PHP en een database (phpMyAdmin) om alle informatie op een gestructureerde manier op te slaan en te beheren. We gebruiken hiervoor ook html, CSS en Javascript.

(The PHP Group, sd)

1.1.3 Gebruikersfunctionaliteiten

In dit project kregen we het websitetemplate, promotion, als basis voor onze applicatie. Binnen deze website wordt voorzien in een automatische weergave van labels en cursussen, zodat de inhoud overzichtelijk en gestructureerd wordt gepresenteerd. Daarnaast is het de bedoeling dat andere gebruikers de cursussen van de leerlingen snel en eenvoudig kunnen raadplegen.

Verder moet er een inlogsysteem worden ontwikkeld. Via deze inlogpagina kunnen de leerlingen hun cursussen op een gebruiksvriendelijke en overzichtelijke manier beheren. Dit houdt in dat zij hun cursussen kunnen toevoegen, aanpassen en organiseren. Ook wordt er voorzien in accountbeheer, zodat gebruikers hun persoonlijke gegevens en accountinstellingen zelfstandig kunnen wijzigen.

(Webestica, sd)

1.2 Uitvoering

1.2.1 Technische opzet

Voor de uitvoering van dit project starten we met het opzetten van de ontwikkelingsomgeving. We installeerden een lokale server via XAMPP, waarbinnen Apache en MySQL actief zijn. De database werd aangemaakt in phpMyAdmin en bevat tabellen voor de gebruikers, cursussen, oefeningen en labels.

Figuur 2: XAMPP

De structuur van de website is opgebouwd op basis van het opgegeven template Promotion. We pasten de HTML- en CSS-bestanden aan, zodat de stijl aansloot bij het thema van 6SPB. De navigatie werd uitgebreid met een overzichtspagina voor cursussen en een inlogpagina voor leerlingen.

(The Apache Software Foundation, sd)

(phpMyAdmin contributors, sd)

(MariaDB Foundation, sd)

1.2.2 Back-endontwikkeling (PHP & database)

Via PHP werd de connectie met de database en de logica voor het ophalen en weergeven van cursussen en labels geïmplementeerd. Cursussen worden automatisch ingeladen vanuit de database en dynamisch weergegeven op de overzichtspagina. Elke cursus is gekoppeld aan een of meerdere labels die de doelspiergroep aanduiden.

Het inlogsysteem maakt gebruik van sessies in PHP. Na een succesvolle login worden gebruikers doorgestuurd naar hun persoonlijk dashboard, waar ze hun cursussen kunnen toevoegen, bewerken en verwijderen. Wachtwoorden worden gehasht en opgeslagen via password_hash() voor een veilige authenticatie.

(The PHP Group, sd)

1.2.3 Front-end en gebruiksvriendelijkheid

De front-end werd opgebouwd met HTML5, CSS3 en Javascript. De template werd aangepast, zodat de lay-out nu naar onze smaak is. Met Javascript werd interactiviteit toegevoegd, zoals het filteren van cursussen op label en het valideren van formulieren aan de clientzijde. Het sturen van data naar de database werd onder andere ook door Javascript gedaan, door het werken met Ajax-calls.

Extra aandacht ging naar het bewerken van het profiel, waar de gebruikers hun persoonlijke gegeven konden aanpassen. De wijzigingen worden dan direct verwerkt in de database en de gebruiker kan meteen door met werken.

(Mozilla Foundation, sd)

(Bootstrap Team, sd)

1.2.4 Account maken of inloggen

Voor de login pagina maakte ik gebruik van Eduport's template. Hoe het werkt maakte ik zelf. De standaardprocedure om informatie door te sturen naar de back-end is door gebruik te maken van een invul veld, zoals: vul hier je gebruikersnaam in. Daarna kan je via Javascript een variabele koppelen aan de waardes die je hebt ingevuld, bijvoorbeeld: Sem. Vanaf daar sturen we via een Ajax call de informatie door naar een Php bestand dat dan de info veilig in de database opslaat. Waarmee je later kan inloggen.

(Webestica, sd)

1.2.5 Profiel bewerken

Het bewerken van je profiel werkt in principe precies hetzelfde als een account aanmaken. Je vult een veld in, bijvoorbeeld je leeftijd. Dan kan je door het opslaan knopje weer met Javascript de informatie ophalen en doorsturen via een Ajax call. PHP haalt deze waardes weer op en update de database naar wat je had ingevuld.

1.2.6 Maken van fiches en cursussen

Ook werkt het maken van fiches en cursussen op dezelfde manier. Van een invult veld naar de database. Na het versturen van je fiche kan je deze fiches zien op je dashboard, waar je ook je fiche kan verwijderen en later kan bewerken. Nu kan je met deze fiches een cursus maken, die je dan weer kan publiceren en bekijken. Zo kunnen de mensen je cursus volgen.

1.3 Resultaat Project (besluit)

1.3.1 Doelen en behaalde doelen

Een aantal doelen die we wilden bereiken waren het aanmaken van cursussen en fiches, waarbij de fiches versleepbaar moesten zijn. Ook hebben we een inlogsysteem gemaakt waarbij de gebruikers gebruiksvriendelijk kunnen inloggen. Het is ook belangrijk dat de gebruiker zijn eigen info kan aanpassen. Waaronder zijn profielfoto, voornaam, achternaam, leeftijd, gender, lengte en gewicht. Met deze info kunnen we dan uiteindelijk filteren op resultaten die beter geschikt zijn voor de gebruiker zelf. Je kan ook je eigen gemaakte fiches bekijken en verwijderen. Dit zijn alle behaalde doelen.

(The jQuery Foundation, sd)

2 Project - Reactiespel

2.1 Opdracht

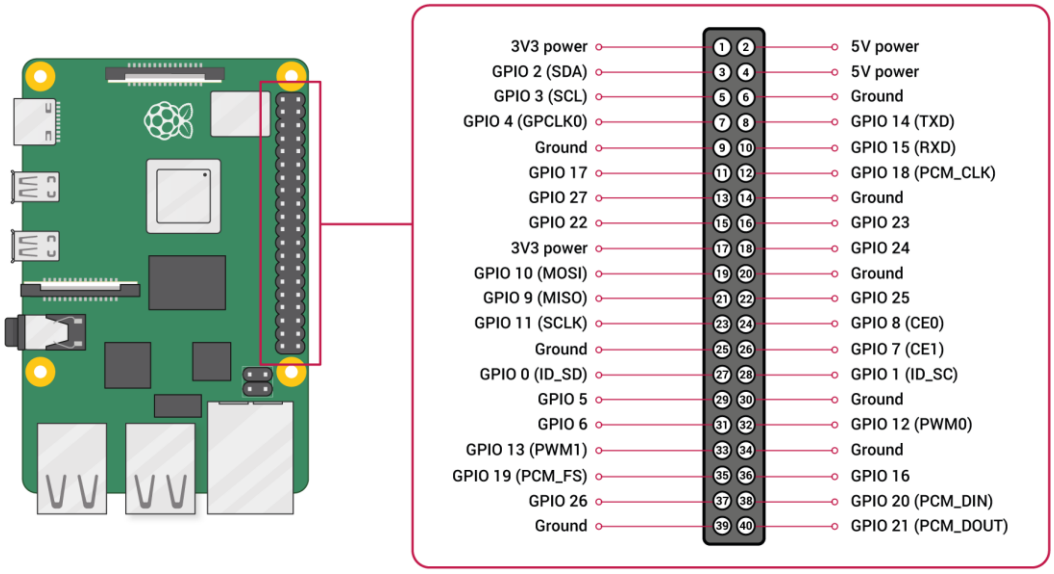
Voor dit project hebben we een reactiespel ontwikkeld waarbij de speler een knop indrukt om een timer te starten. Het doel van het spel is om de knop opnieuw te klikken zodra de timer tien seconden bereikt. De nauwkeurigheid van de speler wordt beoordeeld op basis van hoe dicht de klik bij de tien seconden ligt. Om een competitief element toe te voegen, worden de vijf beste scores opgeslagen op een scorebord, zodat spelers hun prestaties kunnen vergelijken en verbeteren. Dit spel biedt een eenvoudige manier om reactietijd en concentratie te trainen.

2.2 Benodigdheden

- Arcade knop
- Raspberry Pi 4 (model b)
- Scherm, toetsenbord, HDMI, power supply
- Breadboard met 4+ kabeltjes
  (Raspberry Pi Ltd., sd)

2.3 Opbouw

2.3.1 Pinout Raspberry Pi 4



Figuur 3 raspberry Pi mode 4 b

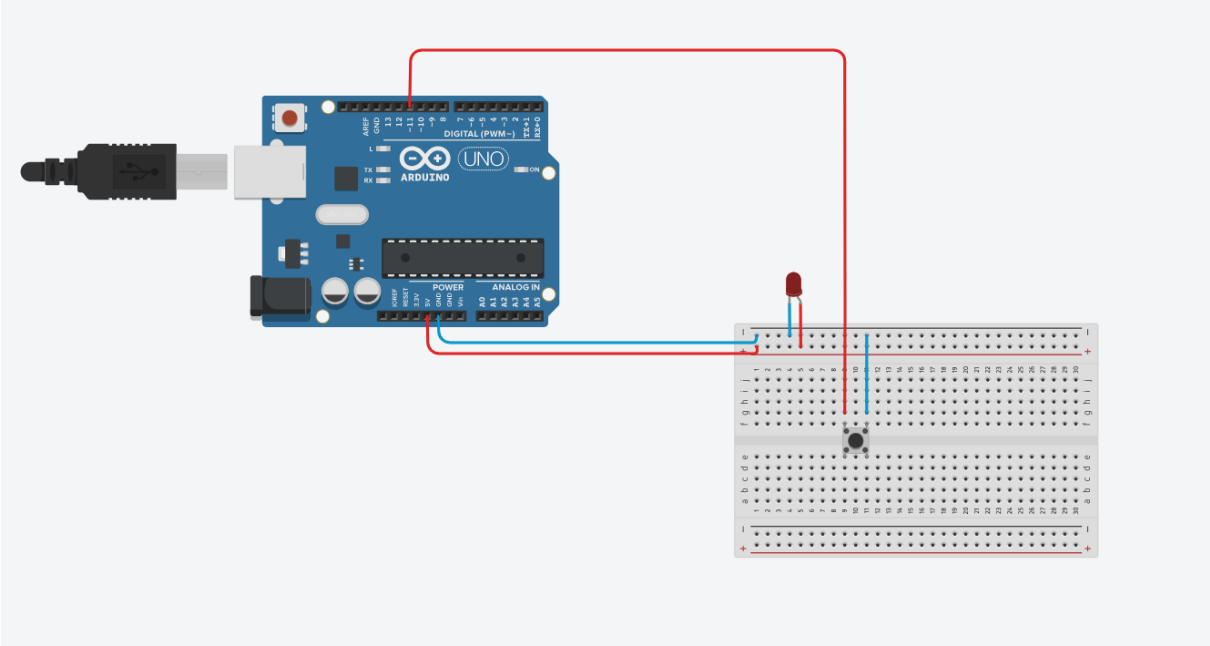
2.3.2 Aansluiten

- Sluit een draad aan van de 5V naar de positieve kant op je breadboard en sluit een draad aan op de ground (gnd) naar de negatieve kant op je breadboard sluit hier je lamp op aan.
- Sluit de zwarte kant van de knop aan op de GPIO 17 (11), en de rode kant aan op de ground.
  Doe dit op je breadboard of doe het rechtstreeks. Zie schema hieronder:

Let op: op de afbeelding staat een Arduino UNO – dit moet de Raspberry Pi (b) zijn.

Ook staan hier gewone led en knop – de led en knop die je nodig hebt is een ledknop in elkaar.

Zie afbeeldingen.



Figuur 4 Arduino uno – aansluiting raspberry en knop

2.4 Raspberry Pi resetten

2.4.1 Benodigdheden

- SD-kaart (min. 8GB)
- SD-kaartlezer (normaal gezien zit dit al in je laptop zelf)
- Computer/laptop
- Raspberry Pi Imager

(Raspberry Pi Ltd., sd)

2.4.2 Reset de SD kaart

1. Steek de SD-kaart in je computer
2. Open **Raspberry Pi Imager**
3. Kies:
   - OS → *Raspberry Pi OS (recommended)*
   - Storage → jouw SD-kaart
4. Klik op (instellingen, optioneel maar handig):

   - Stel **gebruikersnaam + wachtwoord** in
   - Zet **SSH** aan (optioneel, ik doe dit wel)

5. Klik op **Write** (dit wist alles)

2.4.3 Raspberry Pi opstarten

1. Haal de SD-kaart uit je pc
2. Steek hem in je Raspberry Pi
3. Sluit aan:
    - Stroom
    - Scherm + toetsenbord of via SSH (niet aangeraden)
4. Laat hem opstarten

(Raspberry Pi Ltd., sd)

2.4.4 Nieuw account

Als je stap 1 instellingen gebruikte:

- Account is **al aangemaakt** → meteen inloggen

Als je dat niet deed:

- Bij eerste opstart krijg je een **setup wizard**
- Maak daar:
    - Gebruikersnaam
    - Wachtwoord

Moest je SHH wel willen gebruiken kan je best Putty installeren via de officiële website.

2.5 Code

De volgende code wordt uitgevoerd op Thonny in je Raspberry Pi 4.

Let op: je moet een aantal libraries installeren voor dat je dit kan runnen.

Run deze commando's in je CMD op je Raspberry Pi:

- pip install pygame
- sudo apt-get install python3-rpi.gpio

Tip: Voor een Raspberry Pi is het vaak handig om een virtuele omgeving te gebruiken, zodat je libraries zoals pygame en RPi.GPIO netjes geïsoleerd installeert (Dit heb ik zelf niet gedaan):

- python3 -m venv venv
  source venv/bin/activate

  pip install pygame RPi.GPIO

(Pygame community, sd)

(Python Software Foundation, sd)

(Python Software Foundation, sd)

2.6 Toekomst

Er zijn verschillende manieren waarop het reactiespel verder verbeterd kan worden. Zo kan de functionaliteit van de knop uitgebreid worden, bijvoorbeeld door meerdere acties toe te wijzen, zoals het pauzeren van de timer, het terugzetten naar nul of het navigeren door menu's door middel van externe knoppen. Daarnaast kan het spel visueel aantrekkelijker gemaakt worden door animaties of kleurveranderingen bij het indrukken van de knop. Ook kan het scorebord uitgebreid worden met extra statistieken, zoals de gemiddelde fout per speler of het aantal gespeelde rondes. Tenslotte kan het spel uitgebreid worden met verschillende moeilijkheidsgraden, bijvoorbeeld door kortere of langere timers, waardoor het uitdagender wordt voor gevorderde spelers.

□□□□□
□

2.7 Afbeeldingen

3 Project 3 – Web HMI

3.1.1 Inleiding

3.1.2 Wat is dit project?

Stel: je werkt in een fabriek waar je dikke en dunne lagers moet sorteren en verplaatsen. Dit gebeurt door een module die wordt aangestuurd door een plc (Programmable Logic Controller). Een plc is een soort computer die je kan bedienen met fysieke knoppen of met een HMI (Human Machine Interface), een scherm dat vast aan de machine hangt. Dat is heel ouderwets, want je moet er fysiek bij staan om de machine te bedienen. Daarom vervang ik de HMI van Vincent Smetsers door een Web HMI: een website waarmee je de machine kan besturen van waar je ook bent, zolang je maar internet hebt.

In plaats van een scherm aan de muur maak ik een website die hetzelfde kan, maar dan vanop afstand.

3.1.3 Systeemlagen

Het systeem bestaat uit een aantal lagen die met elkaar moeten communiceren. Het is makkelijker om die eerst even op te sommen, want daarna wordt de rest veel duidelijker.

Onderaan heb je de plc van Vincent, de Siemens S7-1215C. Deze machine stuurt de motoren en sensoren aan. De plc kan op zichzelf werken met handmatige besturing of met een HMI.

Het probleem is dat je de plc enkel kan bereiken via het netwerk van school. Want je kan niet zomaar uit dat netwerk zonder beveiligingen uit te schakelen, en dat mochten we niet doen. Daarom hebben we een Raspberry Pi ertussen gezet. De Raspberry Pi zit op hetzelfde netwerk als de plc en kan er rechtstreeks mee praten. De Pi heb ik geprogrammeerd met Python.

Daarboven komt de website. Die moet je kunnen openen vanaf eender welke laptop ter wereld. De volgorde is dus: je laptop, het internet, de Raspberry Pi en de plc. En weer terug.

3.2 Uitvoering

3.2.1 De plc

De plc is het hart van de sturing in Vincents module. Alle componenten zijn

rechtstreeks verbonden met deze controller. De plc verzamelt continu signalen van de sensoren en beslist op basis van de geprogrammeerde logica welke acties uitgevoerd moeten worden.

Dankzij deze centrale sturing verloopt het volledige proces automatisch, zonder dat iemand er met de hand bij moet. De plc zorgt ervoor dat de juiste ventielen op het juiste moment worden aangestuurd, volgens het geprogrammeerde GRAFCET-verloop. Een GRAFCET is een soort stappenschema dat beschrijft welke stap na welke stap komt en onder welke voorwaarde. Daarnaast wordt ook de keuze van de gebruiker via de HMI verwerkt en kunnen veiligheidsacties uitgevoerd worden, zoals een stop bij storing.

De plc die Vincent gebruikt, is de Siemens SIMATIC S7-1215C DC/DC/Relay (6ES7215-1HG40-0XB0). Deze compacte controller beschikt over:

- 14 digitale ingangen (24 V DC)
- 10 digitale uitgangen (relais), geschikt voor hogere schakelstromen
- 2 analoge ingangen (0-10 V DC) en 2 analoge uitgangen (0-20 mA)
- Een ingebouwde PROFINET-interface met 2 poorten voor communicatie
- Ingebouwde high-speed tellers
- Uitbreidbaar met signaalborden, communicatiemodules en tot 8 signaalmodules

De PLC wordt geprogrammeerd met TIA Portal. (Siemens, 6ES7215-1HG40-0XB0, 2026) (Siemens AG, 2026) (Siemens AG, sd)

3.2.2 Wat is een merker, ingang en uitgang?

Voor je verder leest, is het handig om te weten wat ik bedoel met al de adressen zoals Q0.1, I0.3 of M20.1. In een plc heb je drie soorten geheugen die je gebruikt:

Ingangen (I): dit zijn de signalen die binnenkomen, bijvoorbeeld van een sensor. Een ingang kan de plc enkel lezen, niet veranderen. Bijvoorbeeld I0.3 is de sensor die zegt of de uitstoter in rust staat.

Uitgangen (Q): dit zijn de signalen die de plc naar buiten stuurt, bijvoorbeeld om een ventiel of motor aan te sturen. Een uitgang kan de plc zelf hoog of laag zetten. Bijvoorbeeld Q0.2 stuurt de uitstoter aan.

Merkers (M): dit zijn interne geheugenbitjes van de plc. Ze hangen niet vast aan een fysieke draad, maar de plc gebruikt ze om dingen te onthouden of door te geven, zoals "start het programma" (M20.1) of "er is een storing" (M89.7). Een merkerwoord (MW) is gewoon een merker die een heel getal kan onthouden in plaats van enkel 0 of 1, bijvoorbeeld MW100 voor het aantal cyclussen.

Het punt in een adres zoals Q0.2 betekent: byte 0, bit 2. Dat hoef je niet echt te onthouden, maar het verklaart waarom alles met punten geschreven wordt.

Telkens zeg ik dan: "Zet M20.1 hoog." Dan bedoel ik dat ik de plc het signaal geef om te starten.

3.2.3 Flask API (plc_api.py)

Op de Raspberry Pi draait een programma dat ik geschreven heb in Python met Flask. Flask is een hulpmiddel waarmee je een kleine webserver kan maken. Je kan het zien als een master-slave-situatie: de website (de master) geeft een vraag en de Pi reageert. Officieel is die slave de API.

Mijn Flask-programma heeft verschillende endpoints.

Elk endpoint doet iets anders. De belangrijkste zijn:

- /status – vertelt of de plc verbonden is.
- /config – geeft de hele lijst met knoppen, ingangen, uitgangen en merkers door aan de website, zodat de website weet wat ze moet tonen.
- /plc/<naam> – schrijft een waarde naar de plc, bijvoorbeeld een uitgang aanzetten of een merker pulsen.
- /plc/lees_sensoren – geeft de huidige toestand van alle sensoren terug.
- /alarm/huidig – vertelt of er op dit moment een alarm actief is en welk.
- /slede/drive – stuurt de elektrische slede naar een bepaalde positie.
- /rfid/lees en /rfid/schrijf – lezen of schrijven van een RFID-tag.

Het mooie aan Flask is dat het constant op de achtergrond blijft draaien. Het is zo opgezet dat er meerdere taken tegelijk lopen: één taak die elke 0,1 seconde alle sensoren uitleest, één taak die de verbinding met de plc bewaakt en automatisch herverbindt als die wegvalt, één taak die alarmen in de gaten houdt, en één taak die cyclustijden meet. Zo hoeft de website nooit te wachten en blijft alles vlot werken. (Pallets Projects, sd)

3.2.4 snap7 – communicatieprotocol

Een plc en een Raspberry Pi spreken niet zomaar dezelfde taal. De Siemens-plc gebruikt een eigen protocol (het S7-protocol) om data uit te wisselen. Om daarin in Python bij te kunnen, gebruiken we een hulpmiddel dat snap7 noemt. Je kan Snap7 zien als een vertaler tussen Python en de PLC.

Met Snap7 kan ik rechtstreeks in het geheugen van de plc lezen en schrijven. Dat gaat per geheugengebied: PE voor de ingangen (I), PA voor de uitgangen (Q) en MK voor de merkers (M). Wil ik bijvoorbeeld weten of een sensor hoog staat, dan lees ik de juiste byte uit het PE-gebied en kijk ik naar het juiste bitje. Wil ik een uitgang aanzetten, dan schrijf ik naar het PA-gebied.

Belangrijk detail: omdat er meerdere taken tegelijk met de plc willen praten, kan het mislopen als ze dat op exact hetzelfde moment doen. Daarom heb ik een soort slot ingebouwd (een lock). Telkens als een taak met de plc praat, doet ze het slotje even op het slot, en daarna weer open. Zo kunnen er nooit twee dingen tegelijk gebeuren die elkaar in de war brengen.

(Nardella, sd)

3.2.5 Connectie raspberry pi en plc

De plc is altijd verbonden met het netwerk via een ethernetkabel, net zoals de Raspberry Pi. Het is belangrijk dat die twee op hetzelfde netwerk zitten en blijven. De Raspberry Pi en de plc communiceren dus over hetzelfde netwerk.

De Flask-server op de Pi stelt voortdurend vragen aan de plc via dat netwerk. De vragen die hij stelt zijn bijvoorbeeld: "staat uitgang 1 aan?" of "zet merker 3 hoog". De plc antwoordt op die vragen of doet wat de Pi hem opdraagt. Net zoals elke computer een huisnummer heeft op een netwerk (een IP-adres (internetprotocol)), heeft de plc er ook één. Met dat adres weet de Raspberry Pi precies naar wie hij zijn vragen moet sturen. Voor de veiligheid geef ik het hier niet door.

Als de verbinding met de plc om de een of andere reden wegvalt, bijvoorbeeld omdat de kabel loskomt of de plc heropstart, dan probeert de Pi elke 5 seconden automatisch opnieuw te verbinden. De gebruiker hoeft daar dus niets voor te doen. Van zodra de plc terug bereikbaar is, pikt het systeem de draad weer op en kan je weer verder.

3.2.6 Connectie tussen Raspberry Pi en de Website

De Raspberry Pi staat vast verbonden met de plc. Maar een website moet je kunnen openen vanaf overal. Dat is een probleem, want je kan niet zomaar van buiten het netwerk binnendringen zonder beveiligingen uit te schakelen, en dat wilden en mochten we niet doen. Het schoolnetwerk moet zo veilig mogelijk blijven, dus daar nam ik geen risico mee.

We lossen dit op met een tool genaamd ngrok. De Raspberry Pi maakt verbinding met de servers van ngrok op het internet. Ngrok geeft dan een publiek webadres terug. Wanneer je op de website een knop indrukt, stuurt de website dat naar ngrok. Ngrok stuurt het door via de tunnel naar de Raspberry Pi. De Pi verwerkt het en stuurt een antwoord terug. Ngrok geeft dat antwoord aan de website, en de website toont het resultaat. Zo praat je dus eigenlijk met de plc zonder ooit in het schoolnetwerk te moeten zitten.

3.2.7 Hoe veilig is ngrok?

Het goede nieuws is dat de verbinding versleuteld is via HTTPS. Dat betekent dat iemand die het verkeer onderschept er niets mee kan: hij ziet enkel een hoop onleesbare tekens. Toch zijn er ook een paar zwakke punten. Wie het ngrok-adres kent, zou in theorie de plc kunnen aansturen via bepaalde commando's, zonder de website ooit geopend te hebben. En de gratis versie van ngrok heeft geen

wachtwoord op de tunnel zelf, dus die staat open voor iedereen die het adres kent. Daarom heb ik nog een extra beveiliging ingebouwd met een API-key en gebruikersrollen, die ik verderop uitleg. (ngrok Inc., sd)

3.3 Website & gebruikersinterface

3.3.1 Technische opzet van de website

De website zelf is gemaakt met PHP, HTML, CSS en JavaScript, en gebruikt Bootstrap voor de lay-out. PHP zorgt voor de dingen aan de serverkant, zoals inloggen en het ophalen van gebruikers uit de database. JavaScript zorgt voor alles wat live moet gebeuren in de browser, zoals het indrukken van een knop en het versturen ervan naar de Flask API. Voor het versturen van data gebruik ik fetch-oproepen; dat is een manier om vanuit JavaScript een vraag naar een server te sturen. (Bootstrap Team, sd)

3.3.2 Database

De gebruikersgegevens worden opgeslagen in een MariaDB-database op een externe server. De tabel met gebruikers bevat onder andere de kolommen id (een unieke UUID), username, email, password (gehasht), role en de aanmaakdatum. Elke gebruiker krijgt bij registratie automatisch een unieke UUID v4 als sleutel. De databaseverbinding wordt centraal beheerd in één bestand (dbh.php), zodat alle andere PHP-scripts dezelfde verbinding hergebruiken. De inloggegevens voor die verbinding staan niet meer in de code zelf, maar in een .env-bestand op de server. Zo kan ik het project op GitHub zetten zonder dat er gevoelige gegevens in de code staan.

Naast de gebruikerstabel heb ik twee extra tabellen bijgemaakt: alarm_log en cyclus_log. Die worden automatisch gevuld terwijl de installatie draait, zonder dat iemand daar iets voor hoeft te doen.

De tabel alarm_log slaat elke storing op met het alarmnummer, de stap waarbij het fout ging, de naam van het component, een probleembeschrijving, een oplossing, het tijdstip en of het alarm al opgelost is. Zo bouw ik vanzelf een storingshistoriek op.

De tabel cyclus_log slaat na elke voltooide cyclus het lagertype (dun of dik), de duur in milliseconden en of de cyclus geslaagd was op. Dat gebruik ik dan op de statistiekenpagina om gemiddelden en prestatie-indicatoren te berekenen.

3.3.3 Login & registratie

Gebruikers kunnen een account aanmaken via de registratiepagina. Ze vullen hun gebruikersnaam, e-mailadres en wachtwoord in. Het wachtwoord wordt nooit zomaar opgeslagen, maar gehasht via de ingebouwde PHP-functie password_hash(). Na registratie worden ze automatisch ingelogd. Bij een volgend bezoek loggen ze in op de loginpagina, waar hun gegevens gecontroleerd worden met password_verify(). Bij een geslaagde login wordt een sessie gestart met de id, naam, e-mail en rol van de gebruiker. Wanneer iemand uitlogt via de knop bovenaan, wordt de sessie volledig afgebroken en vernietigd.

3.4 Rolsysteem

3.4.1 Rolsysteem (Admin / Operator / Viewer)

Het systeem werkt met drie gebruikersrollen, opgeslagen als een getal in de database: 0 is Admin, 1 is Operator en 2 is User (een gewone kijker). De rol bepaalt wat je mag zien en doen. Een user ziet enkel het dashboard, de statistieken en de informatiepagina's, maar mag niets bedienen. Een operator mag bovendien het handmatige scherm en de takenplanner gebruiken. Een admin kan daarbovenop nog de adminpagina openen om gebruikers te beheren.

De navigatie past zich automatisch aan op basis van de rol: knoppen en pagina's die je niet mag gebruiken, worden gewoon niet getoond of staan uitgeschakeld. De rol wordt vanuit PHP doorgegeven aan JavaScript, zodat ook de knoppen zich daaraan aanpassen.

3.5 Dashboard & schermen

3.5.1 Dashboard

Het dashboard is de startpagina na het inloggen en toont het automatische bedrijf van de machine. Er zijn twee panelen: "Commando's" en "Machine status". Vanuit Commando's kan een Operator of Admin een START- of STOP-puls sturen naar de plc (merker M20.1 of M20.2). Het paneel met de machinestatus toont een reeks led's die live aangeven of het programma bezig is, of de initiatie geslaagd is, welk lagertype klaarligt en of de drukmeter in orde is. Die LED's worden elke seconde ververst doordat de website automatisch de sensoren opvraagt bij de Flask API. Onderaan staat een feedbackbalk die elke actie bevestigt of een foutmelding toont.

3.5.2 Handmatig scherm

Het handmatige scherm geeft operators en admins individuele controle over elke actuator. Per actuator is er een toggle-schakelaar die de uitgang vast aan of uit zet, en een drukknop die de uitgang enkel hoog houdt zolang je hem ingedrukt houdt. Dat laatste komt overeen met het gedrag op de echte machine. Naast elke rij staan kleine led-indicatoren die de bijbehorende sensoren tonen, zoals "Rust" en "Uit".

De actuatoren die je kan bedienen zijn de uitstoter, grijper, controle, selectielager, draaiarm en het persluchtventiel. De draaiarm heeft aparte knoppen voor links, rechts en een togglebeweging. Onderaan staat een grote storing-led en een knop om het alarm te resetten. Het alarm komt steeds als er iets misloopt. Daarnaast zijn er nog verschillende alarmen die ik zo meteen ga opnoemen.

3.5.3 Elektrische slede

De elektrische slede is een stappenmotor die via de plc aangestuurd wordt. Op het scherm verschijnt een slider met drie posities (0, 1 en 2). Wanneer je een positie kiest, zet de website de juiste combinatie van bits hoog, en geeft daarna een trigger, zodat de slede effectief naar die positie rijdt. Achter de schermen kan de aansturing eigenlijk veel meer dan drie posities aan: de slede werkt met 6 bits,

waarmee je in principe tot 64 verschillende posities zou kunnen kiezen. Voor dit project heb je dit helemaal niet nodig, dus zijn drie posities voldoende.

Naast de positionering zijn er ook jog-knoppen om de slede met de hand te verplaatsen zolang je de knop ingedrukt houdt (links en rechts), een home-knop om de slede naar zijn nulpunt te sturen, een aparte aan/uit voor de stappenmotor en een reset bij storing. De slider is ook bedienbaar met het toetsenbord (pijltjestoetsen) en door te slepen met de muis of op een aanraakscherm.

3.5.4 Takenplanner

De takenplanner laat operators en admins een reeks opdrachten samenstellen die daarna automatisch achter elkaar uitgevoerd worden. Via slepen-en-neerzetten sleep je taken vanuit een bibliotheek naar een volgorde-lijst. De beschikbare taaktypen zijn: dunne lager en dikke lager.

Voor een lager-cyclus verloopt elke taak in vaste stappen. Eerst controleert het systeem of er geen alarm actief is. Als er wel een is, wacht het tot het alarm gereset wordt. Daarna wordt de juiste lagerkeuze ingesteld via de merkers M40.0 (dun) of M40.1 (dik), het automatische programma geactiveerd (M30.1) en de semi-automatische modus uitgeschakeld (M30.2 laag). Dan geeft het systeem een startpuls van 500 milliseconden en wacht het tot M70.0 (programma_bezig) hoog gaat. Dat bevestigt dat de plc de cyclus gestart heeft. Vervolgens wacht het systeem tot M70.0 weer laag gaat, want dat betekent dat de cyclus volledig afgelopen is.

Vroeger wachtte het systeem op M70.1, de markering dat het programma gedaan is. Maar M70.1 is maar een korte puls van ongeveer een seconde. Door de kleine vertraging van de verbinding via ngrok kan die puls gemist worden. Daarom heb ik dit aangepast naar het bewaken van M70.0: die merker blijft hoog gedurende de volledige cyclus, en gaat pas laag op het einde. Dat is veel betrouwbaarder.

Er zitten veiligheidstijden op elke stap: als de cyclus niet op tijd start of niet op tijd klaar is, stopt de reeks met een foutmelding in het log. De operator kan ook op elk moment op stop drukken. Reeksen kunnen worden opgeslagen in de browser, zodat je ze later opnieuw kan laden zonder alles opnieuw samen te stellen.

3.5.5 Statistieken

De statistiekenpagina toont twee soorten cijfers.

Het eerste deel komt rechtstreeks uit de plc. De tellers staan niet in het merkergeheugen maar in een apart databouwblok: DB14. Dat is een apart stuk geheugen in de Siemens plc dat je kan gebruiken om waarden in op te slaan die niet aan een fysiek signaal hangen. Elke teller is een DInt (een geheel getal van 32 bits) en staat op een vast adres in dat blok. Zo staat het totale aantal goede cyclussen op DB14.DBD0, het aantal goede dunne lagers op DB14.DBD4, het aantal goede dikke lagers op DB14.DBD8, het totaal aantal fouten op

DB14.DBD12, enzovoort. De Flask API leest die waarden via snap7 rechtstreeks uit het databouwblok en stuurt ze door naar de website, waar ze elke vijf seconden ververst worden.

Het tweede deel wordt berekend uit de database. Telkens als een cyclus voltooid is, schrijft de Pi de duur automatisch weg in cyclus_log. Met die gegevens bereken ik de gemiddelde cyclustijd (in totaal en apart per type lager), een kwaliteitsscore, de storingsfrequentie, de beschikbaarheid en de OEE. OEE staat voor Overall Equipment Effectiveness en is een bekende maatstaf in de industrie om te zien hoe goed een machine effectief presteert. Als benchmark gebruik ik een theoretische cyclustijd van 30 seconden: dat is de tijd waarbinnen een cyclus normaal zou moeten passen. Onderaan staat ook een grafiek van de laatste cyclustijden, zodat je in een oogopslag ziet of de machine sneller of trager begint te werken.

3.5.6 Diagnose & alarmen

De diagnosepagina is bedoeld om snel te zien wat er misloopt als de machine in storing gaat.

In de plc heeft elk alarm een eigen merkerbit. Alarm 0 zit op M0.0, alarm 1 op M0.1, enzovoort tot alarm 15 op M1.7 en alarm 16 op M2.0. Onderhoud heeft een aparte bit op M2.1. Bovendien zet de plc M89.7 hoog als de alarmlamp moet branden. Op de achtergrond leest de Flask API elke 200 milliseconden die bits uit en kijkt welke als eerste hoog staat. Dat is dan het actieve alarmnummer.
In de code heb ik een lijst van 17 alarmen vastgelegd, elk met een componentnaam, de stap waarbij het kan optreden, een probleembeschrijving en een oplossing. Die lijst ziet er bijvoorbeeld zo uit voor alarm 15 (noodstop): component is SF1, het probleem is dat de noodstop geactiveerd werd, de oplossing is de noodstop ontgrendelen en daarna de alarmreset op de HMI indrukken.

De alarmdetectie werkt op de stijgende flank van M89.7: elke keer dat de alarmlamp opnieuw hoog gaat, wordt het alarm weggeschreven in alarm_log, ongeacht of het hetzelfde alarmnummer is als de vorige keer. Zo mis ik geen enkel alarm, ook al is het maar heel kort actief.

Op de website zelf zie je bovenaan een rode banner als er een alarm actief is, met de uitleg en de oplossing erbij. Daaronder staat een tabel met de volledige alarmgeschiedenis, gesorteerd van nieuw naar oud, met paginering van 20 per pagina. Admins en operators kunnen een alarm als opgelost markeren. Onderaan staat een Pareto-grafiek: een staafdiagram van de vijf alarmen die het vaakst voorkomen, zodat je meteen ziet welk onderdeel het meeste problemen geeft.

3.5.7 RFID

De module bevat ook RFID, een systeem waarbij je informatie op een tag kan lezen en schrijven met een draadloze lezer. Op mijn website heb ik daar een apart paneeltje voor. Met de knop “Lees RFID” laat ik de plc een tag uitlezen, waarna de website toont of er een dun of een dik lager op de tag staat. Met de knoppen “Dun” en “Dik” kan ik omgekeerd ook een lagertype naar een tag schrijven. Achterliggend gebeurt dat door een merker te pulsen die de RFID-actie in de PLC start, en daarna de juiste ingangsbytes uit te lezen of weg te schrijven.

3.5.8 Adminpagina

De adminpagina is enkel zichtbaar voor gebruikers met de rol Admin. Ze toont een tabel met alle geregistreerde gebruikers, met hun id, gebruikersnaam, e-mail, rol en aanmaakdatum. Per gebruiker zijn er twee knoppen: bewerken en verwijderen. Via bewerken kan de admin de rol van een gebruiker aanpassen in een pop-upvenster. Via verwijderen kan hij een account verwijderen, met een bevestigingsvenster eerst. Een admin kan zijn eigen account nooit verwijderen; dat wordt zowel op de website als aan de serverkant geblokkeerd.

3.6 Beveiliging

3.6.1 Serverproxy en API-key

In een eerdere versie stuurde de website rechtstreeks vanuit de browser naar de Flask API. Dat betekende dat de API-sleutel in de JavaScript-bestanden stond, en dus zichtbaar was voor iedereen die de broncode van de pagina bekeek.

Dat heb ik opgelost door een serverproxy te bouwen. De website stuurt nu alle plc-oproepen naar een PHP-bestand op de eigen webserver (plc-proxy.php). Dat PHP-bestand voegt de API-sleutel toe en stuurt het verzoek dan door naar de Flask API op de Pi. De browser ziet de API-sleutel nooit. De sleutel zelf staat in een .env-bestand op de webserver dat niet via de browser bereikbaar is en ook niet op GitHub staat.

3.6.2 Rolgebaseerde toegangscontrole

De beveiliging werkt op twee niveaus. Aan de PHP-kant controleert elke pagina of je wel ingelogd bent en stuurt ze je anders terug naar de loginpagina. Gevoelige onderdelen, zoals de adminpagina, worden enkel getoond als je rol gelijk is aan Admin.

Aan de API-kant stuurt elke oproep ook de rol van de gebruiker mee. De Flask API weigert schrijfopdrachten wanneer die rol gelijk is aan User (de kijkersrol). Zo kan een user, zelfs als hij op de een of andere manier de API-sleutel zou kennen, nog steeds geen uitgang aansturen. Bovendien worden de knoppen op de website al meteen uitgeschakeld voor Users, zodat ze die zelfs niet kunnen indrukken.

3.6.3 CSRF beveiliging

Een CSRF-aanval (Cross-Site Request Forgery) is een aanval waarbij een kwaadaardige website in het geheim een actie uitvoert op jouw naam, omdat je al ingelogd bent op een andere site. Om dat te voorkomen genereert de server bij elke sessie een willekeurige CSRF-token. Die token wordt meegestuurd bij het laden van de pagina en zit in elke POST-aanvraag verstopt in de header. Het PHP-script dat het verzoek ontvangt, vergelijkt die token met de waarde die in de sessie staat. Klopt die niet, dan wordt het verzoek geweigerd. Zo kan een externe site

nooit een geldige actie uitsturen namens een ingelogde gebruiker.

3.6.4 Interne sleutel

Naast de API-sleutel voor de website heb ik een tweede geheime sleutel toegevoegd: de INTERNAL_KEY. Die wordt gebruikt voor de communicatie van de Pi naar de webserver. Telkens als de Pi een alarm of een cyclustijd wil opslaan, stuurt hij die sleutel mee in de header van het verzoek. Het PHP-script dat dat verzoek ontvangt, vergelijkt de sleutel met de waarde uit het .env-bestand. Als die niet overeenkomt, weigert het script het verzoek.
Dit is nodig omdat de endpoints save-alarm.php en save-cyclus.php niet via de normale login werken. De Pi is geen gebruiker die inlogt, dus controleer ik op een andere manier of het verzoek echt van de Pi komt en niet van iemand anders.

3.6.5 HTTPS via ngrok

Alle communicatie tussen de browser en de Flask API verloopt via HTTPS. Ngrok versleutelt het verkeer met een certificaat, waardoor de inhoud van elk bericht onleesbaar is voor iemand die het zou onderscheppen. Dat geldt zowel voor de gewone oproepen als voor de API-sleutel die meegestuurd wordt. De HTTPS-verbinding maakt het ook moeilijk voor iemand om er nepberichten tussen te stoppen.

3.6.6 Wachtwoordbeveiliging (hashing)

Wachtwoorden worden nooit in leesbare tekst opgeslagen. Bij registratie wordt het wachtwoord gehasht met de ingebouwde PHP-functie password_hash(), die het bcrypt-algoritme gebruikt. Bij het inloggen vergelijkt password_verify() het ingevoerde wachtwoord met de opgeslagen hash. Bcrypt voegt automatisch een unieke salt toe, waardoor twee gebruikers met hetzelfde wachtwoord toch een andere hash krijgen. Zo blijven de accounts beschermd, zelfs als de database ooit zou uitlekken.

3.6.7 Zwakke punten & verbeterpunten

Het systeem heeft een aantal bekende zwakke punten waar ik eerlijk over wil zijn.

Ten eerste worden de takenreeksen opgeslagen in de lokale opslag van de browser. Daardoor verdwijnen ze als je een andere browser of een incognito venster gebruikt. Dit opslaan in de database zou dat oplossen, maar dat heb ik niet meer kunnen implementeren.

Ten tweede valideert de PHP-proxy geen inhoud van wat er doorgestuurd wordt. Hij controleert of je ingelogd bent en of de CSRF-token klopt, maar hij stuurt het verzoek dan gewoon door. Een extra controle op toegelaten adressen of waarden zou de veiligheid nog verbeteren.

Ten derde staat het IP-adres van de plc zichtbaar in de top-balk van de website (192.168.152.111). Dat is puur voor de gebruiker informatief bedoeld, maar in een productieopstelling zou je dat beter verbergen.

Ten slotte maakt de proxy gebruik van een ngrok-tunnel met een gratis account. Dat betekent een beperkt aantal verzoeken per minuut en een adres dat verandert bij elke herstart. Voor een echte productieomgeving zou een vaste server met een eigen domeinnaam en een geldig SSL-certificaat een veel betere keuze zijn. Nu wordt de SSL-certificaatvalidatie zelfs uitgeschakeld in de proxy omdat ngrok een zelf ondertekend certificaat gebruikt, wat betekent dat de verbinding technisch niet volledig geverifieerd is.

(OWASP Foundation, sd)

3.7 Resultaat (besluit)

3.7.1 Behaalde doelen

Het doel van dit project was een volledig werkende Web HMI bouwen waarmee de Siemens S7-1215C van Vincent Smetsers vanop afstand bediend kan worden. Dat doel is bereikt. De website laat toe om via een beveiligde tunnel de machine op te starten, te stoppen en handmatig te bedienen, en dat werkt wereldwijd via ngrok.

Bovenop de basisbesturing zijn er nog heel wat extra functies bijgekomen die het systeem echt bruikbaar maken: een rolsysteem, zodat niet iedereen evenveel rechten heeft, een takenplanner die complete sorteercyclussen automatisch afwerkt, een elektrische slede die precies naar een positie gestuurd kan worden, een statistiekenpagina met livecijfers en zelfs OEE, een diagnosepagina die alle storingen bijhoudt en in een grafiek toont, en een RFID-functie om lagertypes op tags te lezen en te schrijven. De cyclustijden en alarmen worden automatisch in een database bewaard, zodat er een echte geschiedenis opgebouwd wordt.

3.7.2 Uitbreidingsmogelijkheden

Er zijn een aantal dingen die ik niet meer heb kunnen implementeren of die het systeem nog beter zouden maken.

Ten eerste zouden de takenreeksen beter opgeslagen worden in de database in plaats van in de browser. Nu verdwijnen ze als je een ander toestel gebruikt. Met een extra tabel in MySQL en een laad- en opslaanknop op de pagina zou je reeksen kunnen delen tussen gebruikers en toestellen.

Ten tweede zou een vaste server met een eigen domeinnaam ngrok volledig overbodig maken. Nu verandert het adres bij elke herstart van de Pi en is er een limiet op het aantal verzoeken per minuut. Met een eigen server valt die beperking volledig weg en is de verbinding ook sneller.

Ten derde zou je de statistiekenpagina kunnen uitbreiden met een exportfunctie. Nu kan je de cyclustijden en alarmen alleen op de website bekijken. Een knop om alles als Excel- of CSV-bestand te downloaden zou handig zijn voor een onderhoudstechnicus die de gegevens wil analyseren.

Ten slotte zou je een e-mail- of sms-melding kunnen toevoegen als er een alarm optreedt. Nu moet je de diagnosepagina zelf openen om te zien of er iets

misloopt. Met een automatische melding weet de operator het meteen, ook als hij niet actief naar de website kijkt.

4 Project 4 – led strip

4.1 Inleiding

4.1.1 Limburg STEM't af

Voor Limburg STEM't af moesten we dit jaar een slim speeltoestel maken. Na een aantal pogingen om een partner te vinden die ons kon helpen bij dit project hadden we er geen gevonden. Zonder partner was het moeilijk om echt een – tussen aanhalingstekens – een slim speeltoestel te maken. Daarom waren we begonnen met een reactie spel, als inleiding voor dit project. (provincie Limburg, 2026)

4.1.2 Inleiding led strip spel

Na een tijdje begon de tijd erg te drukken dus moesten we snel iets doen. Meneer Meyers bestelde hiervoor een aantal led strips van ca. vijf meter. Met deze led strips mochten we wat experimenteren. Ikzelf heb hiermee een soort tower defence game van gemaakt. Dit concept bestond al langer, hiermee ben ik aan de slag gegaan.

4.2 Benodigdheden en aansluiting

4.2.1 Benodigdheden

- Arduino UNO met Grove Shield – Grove shield is niet vereist
- 300-LED WS2812B led strip (externe 5V voeding vereist)
- 3 gekleurde arcade-knoppen: geel, blauw, en rood
- 3 Arduino Grove jumper kabels en 2 arduino kabels
- PC met Python 3 en de bibliotheken pyserial, tkinter

4.2.2 Aansluiten arduino UNO & knoppen

Sluit een grove shield aan op de arduino UNO. Sluit ook je usb-b aan op je arduino en sluit usb-a aan je laptop. Nadat je dit hebt gedaan sluit je je jumper kabels op D2, D4 en D6. De jumper kabels sluit je als volgt aan de arcade knoppen aan: je hebt 4 kleuren, geel en zwart moeten aan de knop, rood en wit aan de led. Sluit D2 aan op de rode knop, D4 op de blauwe en D6 op de gele knop.

Let op: als je geen weerstanden gebruikt, zorg dan zeker dat je in je code een INPUT_PULLUP toevoegt zodat je geen kortsluiting creëert. (Arduino LLC, sd) (Seeed Technology Co., Ltd., sd)

4.2.3 Aansluiten van led strip

De Led strip heeft twee bruikbare kanten, je moet hem maar langs één kant aansluiten. Er zijn vijf verschillende draden: rood en wit voor de externe voeding (wit aan de min, rood aan de plus). De andere drie draden: wit, groen en rood sluit je aan de arduino. De witte aan de GND en de groene aan pin 8. De rode hoeven wij niet te gebruiken. (FastLED contributors, sd)

4.3 Uitvoering

4.3.1 Wat is een tower defence game?

Een tower defence, of letterlijk vertaald een toren verdediging, is een spel waarbij je tegenstanders die langs één kant komen probeert te verwijderen. Dit kan je doen op verschillende manieren. Ik heb gekozen om ze een kleurcode te geven, waarbij je een gele, blauwe en rode tegenstander hebt. Om ze te verwijderen moet je een kogel vuren langs te andere kant die dezelfde kleur heeft als de tegenstander – in volgorde. Stel er komt een blauwe, een rode en nog een blauwe druk je blauw, rood, blauw waarmee een kleur schiet over de led strip waarbij de tegenstander verwijderd zal worden.

4.3.2 Hoe heb ik dit gedaan?

Ik ben tewerk gegaan met de led strip die meneer Meyers voor ons kocht. De led stip komt met een externe voeding, het is belangrijk dat je deze aansluit, zodat je de arduino niet belast. Ik ben dan begonnen met het maken van testjes, zoals gewoon een wave van licht te sturen door de led strip. Ik had al een paar test versies, maar ik ben begin mei opnieuw begonnen met het maken van een single tower defence spel. Dit heb ik daarna verwerkt tot een meer kleuren spel.

4.3.3 Het officiële spel

De led strip stelt het speelveld voor. De shooter staat altijd op positie tien. De vijanden verschijnen aan het einde van de led strip (positie 299) en bewegen stap voor stap richting de shooter. Elke vijand heeft een kleur: geel, blauw of rood. De speler moet de knop van de overeenkomstige kleur indrukken om een vijand te verwijderen. Schiet je met de verkeerde kleur, dan flitst de vijand wit maar blijft

hij leven. Als een vijand de spelerspositie bereikt verlies je een leven. Je begint altijd met drie levens. Het spel wordt steeds moeilijker elke ronde (wave).

4.4 Code en scorebord

4.4.1 Python tkinter

Het spel maakt ook gebruik van een scorebord. Dit score bord wordt getoond op het beeldscherm van je computer. Het scorebord geeft ook je levens weer, de hoogst behaalde score en het aantal kogels dat je kan vuren per kleurgroep. Het scorebord is gemaakt met python, maakt gebruik van tkinter en moet geactiveerd worden in visual studio code met het commando: python score.py COM3.

Let op: COM3 staat voor de usb poort waar de arduino op aangesloten is. Het kan zijn dat dit een ander getal is, zoals COM8.

De topscores worden opgeslagen in een json bestand.

(Python Software Foundation, sd)

(pyserial contributors, sd)

4.4.2 Mappenstructuur

- led_game_color
  - highscores.json
  - led_game_color.ino
  - score.py

4.5 Besluit

4.5.1 Uitbreidingsmogelijkheden

Om het spel uit te breiden kunnen er nog altijd meer en meer waves toegevoegd worden. Het toevoegen van meer knoppen of interactiviteit tussen meerdere mensen zijn ook mogelijke uitbreidingen. Denk ook aan extra vijandtypes of power-ups die de speler kan verzamelen.

Besluit

Aan het begin van dit jaar kreeg ik niet één groot project, maar vier afzonderlijke uitdagingen voor mijn geïntegreerde proef. Nu alles achter de rug is, kan ik terugkijken op een traject waarin ik mijn kennis van zowel software als hardware heb kunnen uitbreiden en toepassen. In dit besluit overloop ik kort wat ik per project heb gerealiseerd en wat ik uit het geheel heb geleerd.

Met het eerste project, *Promotion*, heb ik een gebruiksvriendelijke website gebouwd waarmee de richting 6SPB hun GIP kan voorstellen. Hier leerde ik hoe ik de front-end en de back-end met elkaar verbind: van een invulveld in de browser, via JavaScript en Ajax, tot een veilige opslag in de database met PHP. Het inlogsysteem, het beheren van fiches en cursussen en het bewerken van het profiel werken allemaal volgens datzelfde principe. De doelen die we vooropstelden, zijn behaald.

Het tweede project, het *reactiespel*, bracht me voor het eerst bij de hardware. Met een Raspberry Pi, een arcadeknop en een stukje Python maakte ik een spel dat reactietijd en concentratie traint, met een scorebord om prestaties te vergelijken. Hier leerde ik vooral hoe software en fysieke componenten samenwerken.

Het derde project, de *Web HMI*, was voor mij het meest uitdagende en tegelijk het meest bevredigende. In samenwerking met Vincent Smetsers verving ik de klassieke HMI van een PLC door een website waarmee je de machine van overal ter wereld kan bedienen. Hierbij kwamen zo goed als alle eerdere vaardigheden samen: de Raspberry Pi als tussenstap, een Flask-API in Python, communicatie met de PLC via snap7, een veilige verbinding via ngrok en een volledige website met rolsysteem, dashboard en beveiliging. Dit project toonde me hoe verschillende systeemlagen moeten samenwerken om één geheel te vormen.

Het vierde project, het spel op een *led strip*, sloot het jaar af met een Arduino UNO en een tower-defence-spel, uitgewerkt in Python en C++. Ook hier kwamen hardware en programmeren opnieuw samen, maar in een speelsere vorm.

Wanneer ik op het geheel terugkijk, ben ik tevreden met het eindresultaat. De projecten verschilden sterk van elkaar, maar overlapten toch regelmatig in hun opbouw, en dat maakte dat ik aangeleerde technieken telkens opnieuw kon inzetten in een nieuwe context. Niet alles verliep vlekkeloos: vooral bij de Web HMI moest ik onderweg bijsturen, en ik zie nog genoeg uitbreidingsmogelijkheden in elk project. Toch zijn de doelen die ik vooropstelde, behaald. Het belangrijkste dat ik dit jaar heb geleerd, is hoe ik de theorie uit de lessen kan omzetten in werkende projecten, en hoe ik een probleem stap voor stap kan aanpakken, zelfs wanneer het in het begin onoverzichtelijk lijkt.

Literatuurlijst

Arduino LLC. (s.d.). *Arduino UNO Rev3 documentatie*. Tratto da https://docs.arduino.cc/hardware/uno-rev3/

Bootstrap Team. (s.d.). *Bootstrap — de meest populaire HTML, CSS en JS bibliotheek*. Tratto da https://getbootstrap.com/docs/

FastLED contributors. (s.d.). *FastLED bibliotheek voor WS2812B en andere LED strips*. Tratto da https://fastled.io

MariaDB Foundation. (s.d.). *MariaDB documentatie*. Tratto da https://mariadb.org/documentation/.

Mozilla Foundation. (s.d.). *MDN Web Docs: HTML, CSS en JavaScript referentie*. Tratto da https://developer.mozilla.org

Nardella, D. (s.d.). *python-snap7: een Python wrapper voor de snap7 bibliotheek*. Tratto da https://python-snap7.readthedocs.io

ngrok Inc. (s.d.). *ngrok documentatie*. Tratto da https://ngrok.com/docs

OWASP Foundation. (s.d.). *OWASP Top Ten — beveiligingsrisico's voor webapplicaties*. Tratto da https://owasp.org/www-project-top-ten/

Pallets Projects. (s.d.). *Flask — een lichtgewicht WSGI web applicatie framework*. Tratto da https://flask.palletsprojects.com

phpMyAdmin contributors. (s.d.). *phpMyAdmin — beheer MySQL/MariaDB over het web*. Tratto da https://www.phpmyadmin.net

provincie Limburg. (2026). *Limburg stem't af*. Tratto da https://limburgstemtaf.be/

Pygame community. (s.d.). *Pygame documentatie*. Tratto da https://www.pygame.org/docs/

pyserial contributors. (s.d.). *pyserial documentatie*. Tratto da https://pyserial.readthedocs.io

Python Software Foundation. (s.d.). *Python 3 documentatie*. Tratto da https://docs.python.org/3/

Python Software Foundation. (s.d.). *tkinter — Python interface voor Tcl/Tk*. Tratto da https://docs.python.org/3/library/tkinter.html

Python Software Foundation. (s.d.). *venv — virtuele omgevingen aanmaken*. Tratto da https://docs.python.org/3/library/venv.html

Raspberry Pi Ltd. (s.d.). *Raspberry Pi 4 Model B*. Tratto da https://www.raspberrypi.com/products/raspberry-pi-4-model-b/

Raspberry Pi Ltd. (s.d.). *Raspberry Pi Imager*. Tratto da https://www.raspberrypi.com/software/

Raspberry Pi Ltd. (s.d.). *Raspberry Pi OS*. Tratto da https://www.raspberrypi.com/software/operating-systems/

Seeed Technology Co., Ltd. (s.d.). *Base Shield V2 — Grove Shield voor Arduino*. Tratto da https://wiki.seeedstudio.com/Base_Shield_V2/

Siemens AG. (2026). *SIMATIC S7-1200 CPU 1215C DC/DC/Relay (6ES7215-1HG40-0XB0)*. Tratto da https://mall.industry.siemens.com

Siemens AG. (s.d.). *TIA Portal — Totally Integrated Automation Portal*. Tratto da https://www.siemens.com/tia-portal

The Apache Software Foundation. (s.d.). *XAMPP — Apache + MariaDB + PHP + Perl*. Tratto da https://www.apachefriends.org

The jQuery Foundation. (s.d.). *jQuery UI — Sortable*. Tratto da https://jqueryui.com/sortable/

The PHP Group. (s.d.). *password_hash — PHP manual*. Tratto da https://www.php.net/manual/en/function.password-hash.php

The PHP Group. (s.d.). *PHP manual*. Tratto da https://www.php.net/manual/en/

Webestica. (s.d.). *Eduport — Education website template*. Tratto da https://eduport.webestica.com/index-4.html

5 Bijlagen

5.1.1 volledige code reactiespel

5.1.2 code led_game_color.ino

5.1.3 code score.py