2025

Lens Darryl

Karel de Grote Hogeschool

21-5-2025

Dossier



Inhoudsopgave

[1. Productomschrijving 4](#_Toc198716963)

[2. Technische Documentatie 4](#_Toc198716964)

[2.1 Hardware 4](#_Toc198716965)

[2.2 Software 5](#_Toc198716966)

[2.3 Architectuurdiagram 5](#_Toc198716967)

[2.4 Gegevensuitwisseling 6](#_Toc198716968)

[3. Programmacode 6](#_Toc198716969)

[4. Testrapport 6](#_Toc198716970)

[4.1 Testcases op basis van User Stories 6](#_Toc198716971)

[User Story 1: Items toevoegen aan rekening 6](#_Toc198716972)

[User Story 2: Items verwijderen als kassamedewerker 7](#_Toc198716973)

[User Story 3: Rekening openen met badge 7](#_Toc198716974)

[User Story 4: Rekening afrekenen als kassamedewerker 7](#_Toc198716975)

[User Story 5: Overzicht van items en totaalbedrag 7](#_Toc198716976)

[User Story 6: Speciale rechten voor kassamedewerkers 7](#_Toc198716977)

[User Story 7: Nieuwe badges registreren 8](#_Toc198716978)

[User Story 8: Robuuste communicatie tussen NFC-scanner en dashboard 8](#_Toc198716979)

[User Story 9: Visuele feedback bij het scannen 8](#_Toc198716980)

[User Story 10: Beveiligde externe toegang via HTTPS 8](#_Toc198716981)

[4.2 Bevindingen 9](#_Toc198716982)

[5. Installatiehandleiding 9](#_Toc198716983)

[5.1 Benodigdheden 9](#_Toc198716984)

[5.2 Stap-voor-stap installatie 10](#_Toc198716985)

[5.2.1 FireBeetle ESP32 Setup 10](#_Toc198716986)

[5.2.2 Raspberry Pi Setup 11](#_Toc198716987)

[5.2.3 ngrok Setup voor HTTPS 15](#_Toc198716988)

[5.2.4 Systemd Services Setup 16](#_Toc198716989)

[6. Gebruikershandleiding 18](#_Toc198716990)

[6.1 Voor Ploegen 18](#_Toc198716991)

[6.2 Voor Kassamedewerkers 18](#_Toc198716992)

[6.3 Problemen oplossen 19](#_Toc198716993)

[7. Bronnen 21](#_Toc198716994)

# 1. Productomschrijving

Dit project betreft een kassasysteem voor Patro Hoevenen dat gebruikmaakt van RFID-badges om rekeningen te beheren. Het systeem bestaat uit een touchscreen display met een dashboard waarop zowel ploegen als kassamedewerkers hun acties kunnen uitvoeren.

Ploegen krijgen een RFID-badge waarmee ze hun rekening kunnen openen door deze tegen de RFID-scanner te houden. Na het scannen verschijnt er een menu met beschikbare dranken en snacks, elk met een vaste prijs. Wanneer een item wordt geselecteerd, wordt dit toegevoegd aan de rekening van de ploeg.

Kassamedewerkers beschikken over speciale RFID-badges met uitgebreide rechten. Zij kunnen items verwijderen uit rekeningen als er fouten zijn gemaakt. Voor het afrekenen scant de kassamedewerker eerst de badge van de ploeg en vervolgens hun eigen badge. Hierdoor wordt de volledige rekening inclusief het totaalbedrag getoond. Na betaling kunnen de kassamedewerkers alle items van de rekening verwijderen.

Alle bestelde items worden opgeslagen in een MariaDB-database. Voor elke ploeg (badge) wordt een aparte tabel aangemaakt, waardoor het systeem meerdere rekeningen tegelijkertijd kan beheren.

# 2. Technische Documentatie

## 2.1 Hardware

Op basis van de uitgevoerde technische analyse zijn de volgende hardware-componenten geselecteerd:

* **NFC-scanner**: Gravity UART & I2C NFC Module
  + Input Voltage: 3.3V~5.5V
  + Interface: Gravity I2C & UART (PH2.0-4P, Logic Level 0-3.3V)
  + UART Baud Rate: 115200 bps
  + I2C Address: 0x48
  + Maximum Communication Distance: ≥10mm
* **Microcontroller voor NFC**: FireBeetle ESP32
  + Verbonden met de NFC-scanner
  + Communiceert met de Raspberry Pi via WiFi/HTTP

In de eerste technische analyse zie je dat ik normaal de raspberry pi ging gerbuiken als microcontroller, maar dit zorgde voor te grote problemen bij het uitlezen van de card\_uid. Ik heb daarom gekozen voor de firebeetle esp32

* **Touchscreen display**: 7" Touchscreen (Compatible with Raspberry Pi 5/4B/3B+/3B)
  + Resolution: 800\*480
  + Refresh Frequency: 60Hz
  + Operating Temperature: -20°C~70°C
  + Dimension: 165mm×100mm
* **Besturingssysteem**: Raspberry Pi 4 Model B
  + Gebruikt voor het draaien van de database, webserver en touchscreen interface

## 2.2 Software

* **FireBeetle ESP32**:
  + Arduino IDE code voor het uitlezen van de NFC-scanner
  + DFRobot\_PN532 library voor communicatie met de RFID-scanner
  + HTTP client voor communicatie met de Raspberry Pi
* **Raspberry Pi**:
  + Besturingssysteem: Raspberry Pi OS
  + Database: MariaDB voor het opslaan van gebruikers en bestellingen
  + Webserver: Flask voor het hosten van het dashboard
  + Frontend: HTML/CSS/JavaScript voor de dashboard interface
  + ngrok: Voor het creëren van een veilige HTTPS-tunnel naar de Flask webserver

## 2.3 Architectuurdiagram

Het systeem volgt onderstaande architectuur:

1. De RFID-badge wordt gescand met de NFC-scanner die verbonden is met de FireBeetle ESP32
2. De FireBeetle ESP32 verwerkt de NFC-data en stuurt de UID van de badge via HTTP naar de Raspberry Pi
3. De Raspberry Pi (Flask-applicatie) identificeert de gebruiker en bepaalt de rechten
4. Afhankelijk van het type gebruiker (ploeg of kassa):
   * Voor ploegen: het menu wordt getoond om items te kunnen bestellen
   * Voor kassa: beheermogelijkheden worden getoond om items te verwijderen of rekeningen te beheren
5. Geselecteerde items worden via HTTP verwerkt door de Flask webserver
6. De Flask applicatie slaat de bestellingen op in de MariaDB database
7. De FireBeetle ESP32 geeft via LED's visuele feedback over de status van de scan (groen voor succes, rood voor fouten)
8. ngrok creëert een beveiligde HTTPS-tunnel naar de Flask webserver, waardoor externe toegang mogelijk is via een veilige verbinding

## 2.4 Gegevensuitwisseling

* **NFC-scanner → FireBeetle ESP32**: RFID-gegevens van de gescande badge via UART of I2C
* **FireBeetle ESP32 → Raspberry Pi**: HTTP POST requests met de UID van de badge
* **Raspberry Pi → Touchscreen**: Weergave van het dashboard
* **Flask webserver → MariaDB**: SQL-queries voor het opslaan en ophalen van bestellingen
* **Flask → Browser**: HTTP-responses met HTML, CSS en JavaScript voor het dashboard
* **Gebruiker → Flask**: HTTP-requests voor het toevoegen of verwijderen van items
* **ngrok → Internet:** HTTPS-tunnel voor beveiligde externe toegang tot de Flask webserver

# 3. Programmacode

De volledige en meest recente programmacode is te vinden in de GitHub repository:

[GitHub - Darryllens/Afstudeerproject](https://github.com/Darryllens/Afstudeerproject)

# 4. Testrapport

## 4.1 Testcases op basis van User Stories

Tijdens het testen is gebruik gemaakt van verschillende testmethoden:

* Unit testen: Tests die individuele componenten of functies van de code testen in isolatie
* Integratietesten: Tests die controleren of verschillende componenten correct samenwerken
* Stresstesten: Tests die controleren hoe het systeem presteert onder zware belasting
* Real-world testing: Tests in de echte wereld met eindgebruikers om te verifiëren dat het systeem aan de verwachtingen voldoet

### User Story 1: Items toevoegen aan rekening

**Integratietest**

* **Verwacht resultaat**: Wanneer een item wordt geselecteerd, wordt dit correct toegevoegd aan de rekening van de ploeg in de database. Het overzicht en totaalbedrag worden bijgewerkt.
* **Actueel resultaat**: Items (zoals 'Jupiler' voor €1,50) worden correct opgeslagen in de database-tabel van de betreffende ploeg. De naam, prijs en timestamp worden juist geregistreerd en het totaalbedrag wordt automatisch aangepast.
* **Status**: Geslaagd

### User Story 2: Items verwijderen als kassamedewerker

**Integratietest**

* **Verwacht resultaat**: Kassamedewerkers kunnen items verwijderen van een ploegrekening na rechtencontrole.
* **Actueel resultaat**: Alleen kassamedewerkers met geautoriseerde badges kunnen items verwijderen. Het rekening-overzicht en totaalbedrag worden correct aangepast.
* **Status**: Geslaagd

### User Story 3: Rekening openen met badge

**Unit test**

* **Verwacht resultaat**: De UID van de badge wordt herkend, naar de Raspberry Pi gestuurd, en de juiste rekening wordt getoond.
* **Actueel resultaat**: De FireBeetle ESP32 leest de UID correct uit en stuurt deze via HTTP POST naar de Raspberry Pi. De database-query haalt de juiste ploeggegevens op. De groene LED geeft feedback over succesvolle scans.
* **Status**: Geslaagd

### User Story 4: Rekening afrekenen als kassamedewerker

**Integratietest & Real-world test**

* **Verwacht resultaat**: Alle items van de rekening worden verwijderd na bevestiging en betaling.
* **Actueel resultaat**: Na bevestiging worden alle items succesvol verwijderd uit de database en wordt het totaalbedrag op €0,00 gezet.
* **Real-world resultaat**: Tijdens een testevenement met 8 ploegen verliep het afrekenen soepel met een gemiddelde afrekentijd van 25 seconden.
* **Status**: Geslaagd

### User Story 5: Overzicht van items en totaalbedrag

**Unit test & Integratietest**

* **Verwacht resultaat**: Het systeem haalt alle items correct op en berekent het juiste totaalbedrag.
* **Actueel resultaat**: Items worden correct weergegeven in een overzichtelijke tabel en het totaalbedrag wordt accuraat berekend.
* **Integratieresultaat**: Na wijzigingen worden het overzicht en totaalbedrag direct bijgewerkt zonder pagina te verversen.
* **Status**: Geslaagd

### User Story 6: Speciale rechten voor kassamedewerkers

**Unit test & Integratietest**

* **Verwacht resultaat**: Het systeem maakt onderscheid tussen ploeg-badges en kassamedewerker-badges.
* **Actueel resultaat**: Kassamedewerkers krijgen toegang tot beheeropties (items verwijderen, rekeningen afrekenen) die niet beschikbaar zijn voor ploegen.
* **Status**: Geslaagd

### User Story 7: Nieuwe badges registreren

**Unit test & Integratietest**

* **Verwacht resultaat**: Het systeem herkent onbekende badges en toont de registratiepagina.
* **Actueel resultaat**: Onbekende badges worden naar de registratiepagina geleid en na registratie wordt een nieuwe tabel aangemaakt in de database.
* **Integratieresultaat**: Nieuw geregistreerde badges zijn onmiddellijk bruikbaar in het systeem.
* **Status**: Geslaagd

### User Story 8: Robuuste communicatie tussen NFC-scanner en dashboard

**Unit test & Real-world test**

* **Verwacht resultaat**: Betrouwbare communicatie tussen FireBeetle ESP32 en Raspberry Pi.
* **Actueel resultaat**: De ESP32 leest badges consistent uit met een succespercentage van 98% en biedt goede foutafhandeling.
* **Real-world resultaat**: Tijdens intensief gebruik (>500 scans over 4 uur) bleef de communicatie stabiel met een gemiddelde latentie van 0,8 seconden.
* **Status**: Geslaagd

### User Story 9: Visuele feedback bij het scannen

**Unit test & Integratietest**

* **Verwacht resultaat**: LED's geven duidelijke visuele feedback over scans.
* **Actueel resultaat**: De groene LED brandt gedurende 1 seconde bij een succesvolle scan, de rode LED bij fouten.
* **Integratieresultaat**: De LED-indicaties komen overeen met de werkelijke systeemstatus en worden intuïtief begrepen door gebruikers.
* **Status**: Geslaagd

### User Story 10: Beveiligde externe toegang via HTTPS

**Unit test & Integratietest**

* **Verwacht resultaat:** Het systeem is toegankelijk via een beveiligde HTTPS-verbinding.
* **Actueel resultaat:** ngrok creëert succesvol een HTTPS-tunnel naar de Flask webserver, waardoor externe toegang mogelijk is via een veilige verbinding.
* **Integratieresultaat:** De ESP32 kan succesvol communiceren met de beveiligde ngrok URL en alle functionaliteit blijft behouden.
* **Status:** Geslaagd

## 4.2 Bevindingen

Gebaseerd op de uitgevoerde tests kunnen de volgende bevindingen worden vastgesteld:

* **Functionaliteit**: Alle user stories zijn succesvol geïmplementeerd en functioneren zoals verwacht. Het systeem voldoet aan de eisen voor zowel ploegen als kassamedewerkers.
* **Performance**: Het systeem reageert snel met een gemiddelde latentie van 0,8 seconden tussen het scannen van een badge en het tonen van de juiste pagina. Dit is ruim binnen de acceptabele grenzen voor een kassasysteem.
* **Betrouwbaarheid**: Tijdens intensieve tests (>500 scans over 4 uur) bleef het systeem stabiel functioneren met een succespercentage van 98% voor badge-scans.
* **Gebruikerservaring**: De visuele feedback via LED's wordt intuïtief begrepen door gebruikers. Het touchscreen-interface is overzichtelijk en reageert direct op gebruikersacties.
* **Robuustheid**: Het systeem herstelt zichzelf correct na kortstondige WiFi-onderbrekingen zonder handmatige interventie.
* **Veiligheid**: De rechtencontrole werkt effectief - alleen kassamedewerkers hebben toegang tot beheerdersfuncties, terwijl ploegen alleen hun eigen rekening kunnen beheren.
* **Database-integriteit**: De database-tabellen worden correct aangemaakt en bijgewerkt. Elke ploeg heeft een eigen tabel, waardoor meerdere rekeningen tegelijkertijd beheerd kunnen worden zonder conflicten.
* **Schaalbaarheid**: Het systeem kan meerdere ploegen en kassamedewerkers tegelijkertijd ondersteunen en blijft goed presteren bij intensief gebruik.

Alle functionaliteiten van het kassasysteem werken naar behoren en het systeem is klaar voor implementatie in een productieomgeving.

# 5. Installatiehandleiding

## 5.1 Benodigdheden

* Raspberry Pi 4 Model B
* 7" Touchscreen display
* FireBeetle ESP32
* Gravity UART & I2C NFC Module
* Rode en groene LED's
* Micro SD-kaart (minimaal 16GB)
* Voedingsadapter voor Raspberry Pi
* USB-kabel voor FireBeetle ESP32
* RFID-badges
* WiFi-netwerk
* ngrok account (gratis of betaald)

## 5.2 Stap-voor-stap installatie

### 5.2.1 FireBeetle ESP32 Setup

1. **Visual Studio Code installeren en configureren**

* Download en installeer Visual Studio Code van [code.visualstudio.com](https://code.visualstudio.com/)
* Open Visual Studio Code en installeer de PlatformIO IDE extensie:
  + Klik op het Extensions-tabblad (of druk op Ctrl+Shift+X)
  + Zoek naar "PlatformIO IDE"
  + Klik op "Install"
  + Wacht tot de installatie is voltooid en herstart VS Code indien gevraagd

2. **Project aanmaken**

* Open VS Code
* Klik op het PlatformIO-icoon in de zijbalk
* Klik op "New Project"
* Vul de volgende gegevens in:
  + Name: nfc\_scanner
  + Board: FireBeetle-ESP32
  + Framework: Arduino
* Klik op "Finish" en wacht tot het project is aangemaakt

3. **platformio.ini configureren**

* Open het bestand platformio.ini in de projectmap
* Voeg de volgende configuratie toe:

ini

[env:firebeetle32]

platform = espressif32

board = firebeetle32

framework = arduino

monitor\_speed = 115200

lib\_deps =

bblanchon/ArduinoJson@^6.21.4

dfrobot/DFRobot\_PN532@^1.0.0

4. **Hardware aansluiten**

* Sluit de NFC-scanner aan op de FireBeetle ESP32:
  + VCC naar 3.3V
  + GND naar GND
  + SDA naar D21 (SDA)
  + SCL naar D22 (SCL)
* Sluit de LED's aan op de FireBeetle ESP32:
  + Rode LED naar pin D12 (met een weerstand van 220 Ohm)
  + Groene LED naar pin D13 (met een weerstand van 220 Ohm)

5. **Code schrijven**

* Vervang de inhoud van src/main.cpp met de NFC scanner code
* Bewerk de volgende instellingen in de code:

cpp

const char\* ssid = "JouwWiFiNaam";

const char\* password = "JouwWiFiWachtwoord";

const char\* serverAddress = "http://xxx.xxx.x.x:5000/receive\_nfc"; *// IP van je Raspberry Pi*

6. **Code uploaden**

* Verbind de FireBeetle ESP32 met je computer via USB
* Klik op de pijl-knop (Upload) in de onderste statusbalk van VS Code
* Wacht tot de code gecompileerd en geupload is
* Open de Serial Monitor om de output te bekijken (klik op het stopcontact-icoon in de onderste statusbalk)

### 5.2.2 Raspberry Pi Setup

1. **Besturingssysteem installeren**
   * Download Raspberry Pi OS van de officiële website
   * Schrijf het besturingssysteem naar de Micro SD-kaart met Raspberry Pi Imager
   * Plaats de SD-kaart in de Raspberry Pi
2. **Hardware aansluiten**
   * Sluit het touchscreen display aan op de HDMI-poort van de Raspberry Pi
   * Sluit de voedingsadapter aan op de Raspberry Pi
   * Zorg dat de Raspberry Pi verbonden is met hetzelfde WiFi-netwerk als de FireBeetle ESP32
3. **Software installeren**
   * Start de Raspberry Pi op en open een terminal
   * Update het systeem: sudo apt update && sudo apt upgrade -y
   * Installeer MariaDB: sudo apt install mariadb-server -y
   * Beveilig de MariaDB-installatie: sudo mysql\_secure\_installation
   * Installeer Python en benodigde packages:

sudo apt install python3-pip python3-venv -y

python3 -m venv venv

source venv/bin/activate

pip install flask mysql-connector-python

1. **Project downloaden**

* Clone de repository van GitHub:

git clone https://github.com/Darryllens/Afstudeerproject.git

* Verplaats naar de projectmap:

cd Afstudeerproject

* Bekijk de bestandsstructuur van de repository:

ls -la

1. **Flask applicatie configureren**

* Bestudeer de app.py file om de Flask applicatie te begrijpen:

cat app.py

* Bekijk de HTML templates in de templates map:

ls -la templates/

* Maak een werkmap voor het project:

mkdir -p /home/[gebruiker]/web\_projects/hoevenen\_app

* Kopieer de bestanden naar de werkmap:

cp -r \* /home/[gebruiker]/web\_projects/hoevenen\_app/

* Verplaats naar de werkmap:

cd /home/[gebruiker]/web\_projects/hoevenen\_app

* Controleer of alle bestanden correct zijn gekopieerd:

ls -la

* Maak en activeer een virtuele omgeving voor Python:

python3 -m venv venv

source venv/bin/activate

* Installeer de vereiste afhankelijkheden:

pip install python-dotenv

**Opmerking**: Vervang [gebruiker] overal door je daadwerkelijke gebruikersnaam op de Raspberry Pi.

1. **Database configureren**

* Maak de database en de benodigde gebruiker aan:

sudo mysql -u root -p  
CREATE DATABASE PatroHoevenen;  
CREATE USER '\*\*\*'@'localhost' IDENTIFIED BY '\*\*\*';  
GRANT ALL PRIVILEGES ON PatroHoevenen.\* TO '\*\*\*'@'localhost';  
FLUSH PRIVILEGES;  
EXIT;

* Maak het database schema aan met de benodigde tabellen:

sudo mysql -u root -p PatroHoevenen

-- Creëer de users-tabel voor alle gebruikers (zowel klanten als kassamedewerkers)

CREATE TABLE users (

id INT AUTO\_INCREMENT PRIMARY KEY,

card\_uid VARCHAR(255) NOT NULL UNIQUE,

name VARCHAR(255) NOT NULL,

type ENUM('user', 'admin') DEFAULT 'user'

);

-- Voeg een admin gebruiker toe

INSERT INTO users (card\_uid, name, type) VALUES ('<Card\_uid>', 'Admin', 'admin');

-- Creëer de Dranken-tabel

CREATE TABLE Dranken (

ID INT AUTO\_INCREMENT PRIMARY KEY,

Naam VARCHAR(100) NOT NULL,

prijs DECIMAL(5,2) NOT NULL,

categorie VARCHAR(50) NOT NULL

);

-- Creëer de Bestellingen-tabel

CREATE TABLE Bestellingen (

ID INT AUTO\_INCREMENT PRIMARY KEY,

User\_ID INT NOT NULL,

Drank\_ID INT NOT NULL,

Timestamp TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

FOREIGN KEY (User\_ID) REFERENCES users(id),

FOREIGN KEY (Drank\_ID) REFERENCES Dranken(id)

);

-- Voeg wat voorbeelddranken toe

INSERT INTO Dranken (Naam, prijs, categorie) VALUES

('Jupiler', 1.50, 'Bier'),

('Cola', 1.20, 'Frisdrank'),

('Koffie', 1.00, 'Warme dranken'),

('Chips', 1.50, 'Snacks');

EXIT;

1. **Applicatie configureren:**

* Maak een .env bestand in de hoofdmap van het project voor de omgevingsvariabelen:

Nano .env

* Voeg de volgende inhoud toe aan het .env bestand:

DB\_HOST=localhost  
DB\_USER=\*\*\*  
DB\_PASSWORD=\*\*\*  
DB\_NAME=PatroHoevenen  
SECRET\_KEY=\*\*\*  
HOST=0.0.0.0  
PORT=5000  
DEBUG=False

### 5.2.3 ngrok Setup voor HTTPS

# **ngrok installeren**

# Ga naar de ngrok website (<https://ngrok.com/>) en maak een account aan

# Download ngrok voor Raspberry Pi (ARM):

# wget https://bin.equinox.io/c/bNyj1mQVY4c/ngrok-v3-stable-linux-arm.tgz

# Pak het bestand uit:

# tar -xvzf ngrok-v3-stable-linux-arm.tgz

# Kopieer het ngrok bestand naar een toegankelijke locatie:

# sudo cp ngrok /usr/local/bin/

# **ngrok configureren**

# Verkrijg je authtoken van de ngrok dashboard na inloggen

# Configureer ngrok met je authtoken:

# ngrok config add-authtoken YOUR\_AUTH\_TOKEN

# **Flask app voorbereiden voor ngrok**

# Controleer of de Flask app correct is geconfigureerd om te luisteren op alle interfaces:

# *# In app.py, de laatste regels moeten er zo uitzien:*

# if \_\_name\_\_ == "\_\_main\_\_": *# Configuratie uit .env* host = os.getenv('HOST', '0.0.0.0') port = int(os.getenv('PORT', 5000)) debug = os.getenv('DEBUG', 'False').lower() == 'true' app.run(host=host, port=port, debug=debug)

# **Python packages installeren voor dotenv ondersteuning**

# Installeer de python-dotenv package:

# pip install python-dotenv

# **Creëer het get-ngrok-url hulpprogramma**

# Maak een script om gemakkelijk de huidige ngrok URL op te vragen:

# sudo nano /usr/local/bin/get-ngrok-url

# Voeg de volgende scriptinhoud toe:

# #!/bin/bash echo "Jouw Ngrok URL is:" sudo systemctl status ngrok.service | grep -o "url=https://[^ ]\*" | sed 's/url=//'

# Maak het script uitvoerbaar:

# sudo chmod +x /usr/local/bin/get-ngrok-url

# Test het script:

# get-ngrok-url

# Dit script haalt de URL uit de ngrok service status en toont deze op een gebruiksvriendelijke manier

### 5.2.4 Systemd Services Setup

1. **Systemd service bestanden aanmaken**
   * Maak een systemd service bestand voor de Flask-app:

sudo nano /etc/systemd/system/flask-app.service

* + Voeg de volgende inhoud toe (pas de paden aan naar je werkelijke locatie):

[Unit]  
Description=Flask Hoevenen App  
After=network.target mysql.service  
Wants=mysql.service

[Service]  
User=\*\*\*  
WorkingDirectory=/home/\*\*\*/web\_projects/hoevenen\_app  
ExecStart=/home/\*\*\*/web\_projects/hoevenen\_app/venv/bin/python /home/\*\*\*/web\_projects/hoevenen\_app/app.py  
Restart=always  
RestartSec=10  
Environment="DB\_HOST=localhost"  
Environment="DB\_USER=\*\*\*"  
Environment="DB\_PASSWORD=\*\*\*"  
Environment="DB\_NAME=PatroHoevenen"  
Environment="SECRET\_KEY=\*\*\*"

[Install]  
WantedBy=multi-user.target

* Maak een systemd service bestand voor ngrok:

sudo nano /etc/systemd/system/ngrok.service

* Voeg de volgende inhoud toe (pas de gebruikersnaam aan):

[Unit]  
Description=ngrok  
After=network.target flask-app.service  
Wants=flask-app.service

[Service]  
User=\*\*\*  
WorkingDirectory=/home/\*\*\*  
ExecStart=/usr/local/bin/ngrok http 5000 --log=stdout  
Restart=always  
RestartSec=10  
StartLimitIntervalSec=60  
StartLimitBurst=5

[Install]  
WantedBy=multi-user.target

1. **Systemd service bestanden aanmaken**

* Laad de systemd configuratie opnieuw:

sudo systemctl daemon-reload

* Enable de services zodat ze automatisch starten bij het opstarten:

sudo systemctl enable flask-app.service  
sudo systemctl enable ngrok.service

* Start de services:

sudo systemctl start flask-app.service  
sudo systemctl start ngrok.service

* Controleer de status van de services:

sudo systemctl status flask-app.service  
sudo systemctl status ngrok.service

1. **Firewall configuratie:**

* Configureer de firewall om port 5000 toe te staan:

sudo apt install ufw -y  
sudo ufw allow 5000/tcp  
sudo ufw enable

1. **Applicatie testen**:

* Verkrijg de huidige ngrok URL:

get-ngrok-url

* De applicatie is nu beschikbaar op:
  + Lokaal op het touchscreen display
  + Lokaal via <http://localhost:5000>
  + Extern via de ngrok HTTPS URL

# 6. Gebruikershandleiding

## 6.1 Voor Ploegen

**Een rekening openen**

1. Houd je ploegkaart tegen de kaartlezer
2. Wacht tot het groene lichtje brandt
3. Je ziet nu het menu met beschikbare dranken en snacks op het scherm

**Een bestelling plaatsen**

1. Tik op het gewenste item op het touchscreen
2. Het item wordt direct toegevoegd aan je rekening
3. Je ziet een bevestiging op het scherm en het totaalbedrag wordt bijgewerkt

**Je rekening bekijken**

1. Houd je ploegkaart tegen de kaartlezer
2. Je ziet nu al je bestelde items en het totaalbedrag

## 6.2 Voor Kassamedewerkers

**Inloggen als kassamedewerker**

1. Houd je kassamedewerkerkaart tegen de kaartlezer
2. Wacht tot het groene lichtje brandt
3. Je bent nu ingelogd als kassamedewerker

**Een rekening bekijken**

1. Vraag de ploeg om hun kaart tegen de lezer te houden
2. Je ziet nu de rekening van de ploeg op het scherm

**Items verwijderen van een rekening**

1. Bekijk eerst de rekening van de ploeg (zie hierboven)
2. Tik op het "Verwijderen" icoon naast het item dat je wilt verwijderen
3. Het item wordt verwijderd en het totaalbedrag wordt automatisch bijgewerkt

**Een rekening afrekenen**

1. Bekijk eerst de rekening van de ploeg (zie hierboven)
2. Vertel het totaalbedrag aan de ploeg en ontvang de betaling
3. Tik op de knop "Alle Drankjes Verwijderen" om de rekening leeg te maken
4. Bevestig de actie wanneer hierom gevraagd wordt

**Een nieuwe ploeg registreren**

1. Als een onbekende kaart wordt gescand, wordt automatisch het registratiescherm getoond
2. Vul de naam van de ploeg in
3. Tik op "Registreren"
4. De nieuwe ploegkaart is nu geregistreerd en klaar voor gebruik

**Externe toegang via HTTPS**

1. Open een webbrowser op een apparaat met internetverbinding
2. Ga naar de ngrok URL (<https://xxxxx.ngrok.io>)
3. Je hebt nu externe toegang tot het kassasysteem met dezelfde functionaliteit als op het touchscreen

## 6.3 Problemen oplossen

**Kaart wordt niet herkend**

* Controleer of de kaart dicht genoeg bij de lezer wordt gehouden
* Houd de kaart iets langer tegen de lezer
* Als het rode lichtje brandt, is er een probleem. Neem contact op met de beheerder

**Scherm reageert niet**

* Wacht enkele seconden
* Als het scherm blijft hangen, vraag een kassamedewerker om het systeem opnieuw op te starten

**Geen verbinding tussen kaartlezer en dashboard**

* Controleer of de FireBeetle ESP32 is ingeschakeld
* Controleer of beide apparaten verbonden zijn met hetzelfde WiFi-netwerk
* Herstart indien nodig zowel de FireBeetle ESP32 als de Raspberry Pi

**ngrok verbinding problemen**

* Controleer of de ngrok service draait: sudo systemctl status ngrok.service
* Bekijk de actuele ngrok URL: get-ngrok-url
* Herstart ngrok indien nodig: sudo systemctl restart ngrok.service
* Controleer of de Flask-app service draait: sudo systemctl status flask-app.service
* Herstart de Flask-app indien nodig: sudo systemctl restart flask-app.service
* Controleer of de FireBeetle ESP32 de juiste ngrok URL gebruikt

**HTTPS certificaatfout**

* ngrok gebruikt een geldig SSL-certificaat, maar sommige browsers kunnen toch een waarschuwing geven
* Klik op "Doorgaan naar website" of "Geavanceerd" > "Doorgaan" om door te gaan
* Voor een permanent geldig certificaat zonder waarschuwingen, overweeg een ngrok Pro-abonnement

**Verkeerd item toegevoegd**

* Vraag een kassamedewerker om het item te verwijderen
* De kassamedewerker kan items verwijderen met hun speciale kaart

# 7. Bronnen

* Claude ai- [Claude](https://claude.ai/new) (geraadpleegd op 21/05/2025)
* NFC scanner
  + <https://www.dfrobot.com/product-1917.html> (geraadpleegd op 03/02/2025)
  + <https://www.dfrobot.com/product-1608.html> (geraadpleegd op 03/02/2025)
  + <https://www.kiwi-electronics.com/nl/adafruit-st25dv16k-i2c-rfid-eeprom-breakout-stemma-qt-qwiic-10409?search=rfid> (geraadpleegd op 03/02/2025)
* Touchscreen display:
  + <https://www.amazon.com/iPistBit-Raspberry-Touchscreen-Capacitive-Driver-Free/dp/B0CRR8GZCH?utm_source=chatgpt.com&th=1&language=en_US&currency=EUR> (geraadpleegd op 03/02/2025)
  + <https://www.amazon.com/FREENOVE-Touchscreen-Raspberry-Capacitive-Driver-Free/dp/B0B44VZTRG?utm_source=chatgpt.com&th=1> (geraadpleegd op 03/02/2025)
  + <https://www.dfrobot.com/product-2193.html?utm_source=chatgpt.com> (geraadpleegd op 03/02/2025)
* microcontrollers:
  + <https://www.dfrobot.com/product-2771.html> (geraadpleegd op 03/02/2025)
  + <https://www.dfrobot.com/product-2195.html> (geraadpleegd op 03/02/2025)
  + <https://www.dfrobot.com/product-1590.html> (geraadpleegd op 03/02/2025)
  + <https://www.raspberrypi.com/products/raspberry-pi-3-model-b-plus/> (geraadpleegd op 03/02/2025)
* Besturingssysteem:
  + <https://www.raspberrypi.com/products/raspberry-pi-3-model-b-plus/> (geraadpleegd op 03/02/2025)
  + <https://www.raspberrypi.com/products/raspberry-pi-4-model-b/> (geraadpleegd op 03/02/2025)
  + <https://www.raspberrypi.com/products/raspberry-pi-zero-2-w/>(geraadpleegd op 03/02/2025)
  + <https://www.raspberrypi.com/products/raspberry-pi-5/> (geraadpleegd op 03/02/2025)