ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

NEUMANN JÁNOS

INFORMATIKAI KAR

SZAKDOLGOZAT

OE-NIK	Hallgató neve:	**Makai Krisztina Katalin**
2023	Hallgató törzskönyvi száma:	**T/006296/FI12904/N**

Óbudai Egyetem
Neumann János Informatikai Kar
Szoftvertervezés és -fejlesztés Intézet

SZAKDOLGOZAT FELADATLAP

Hallgató neve: **Makai Krisztina Katalin**
Törzskönyvi száma: T/006296/FI12904/N
Neptun kódja: [redacted]

A dolgozat címe:

Mobilalkalmazás fejlesztése okos kávéfőző vezérléséhez
Mobile application for smart coffee maker

Intézményi konzulens: Kiss Dániel
Külső konzulens:

Beadási határidő: 2022. december 15.

A záróvizsga tárgyai: Számítógép architektúrák
IoT, beágyazott rendszerek és robotika specializáció

A feladat

A tradicionális kávékészítésnél magasabb minőségi szintet célzó ún. *speciality* kávé mozgalom a 2000-es évek elején indult. Az irányzat követői közül sokan a kávé forrásának igényesebb megválasztása mellett új pörkölési vagy kávékészítési technikák megalkotásával is foglalkozni kezdtek, amely többek között a napjainkban egyre divatosabb *high-end* otthoni eszpresszógépek létrejöttéhez vezetett. Ezeken a gépeken a barista egyéni profilt állíthat be a különféle kávé receptekhez, például automatizálhatja a főzés folyamatát és finomhangolhatja a főzet mennyiségét, a víz hőmérsékletét és nyomását.

A szakdolgozat célja egy olyan okos kávéfőző prototípusának, valamint a vezérléséhez alkalmas mobilalkalmazás megtervezése és elkészítése, amely segítségével a felhasználó által beállított paraméterekkel készíthető a kávé.

A dolgozatnak tartalmaznia kell:

- a megoldandó feladat hátterének rövid összefoglalóját,
- a dolgozat céljának pontos megfogalmazását,
- a hardveres és szoftveres megvalósításhoz alkalmazható eszközök és technológiák ismertetését,
- a kiválasztott hardveres és szoftveres komponensek bemutatását és a választás indoklását,
- az elkészítendő rendszer részletes tervét,
- a megvalósítás menetének dokumentációját,
- az elkészült rendszer tesztelését és az eredmények ismertetését.

ÓBUDAI EGYETEM * Neumann János Informatikai Kar * 1. Budapest

Ph.

.....
Dr. Vámossy Zoltán
intézetigazgató

A szakdolgozat elévülésének határideje: **2024. december 15.**
(OE TVSz 55.§ szerint)

A dolgozatot beadásra alkalmasnak tartom:

.....
külső konzulens

Kiss Dániel
.....
intézményi konzulens

HALLGATÓI NYILATKOZAT

Alulírott hallgató kijelentem, hogy a szakdolgozat saját munkám eredménye, a felhasznált szakirodalmat és eszközöket azonosíthatóan közöltem. Az elkészült szakdolgozatomban található eredményeket az egyetem és a feladatot kiíró intézmény saját céljára térítés nélkül felhasználhatja.

Budapest, 2022.12.15

.....
hallgató aláírása

KONZULTÁCIÓS NAPLÓ

Hallgató neve: Neptun kód: Tagozat:

Makai Krisztina Katalin [redacted] **nappali**

Telefon: Levelezési cím (pl: lakcím):

[redacted]

Szakdolgozat címe magyarul:

Mobilalkalmazás fejlesztése okos kávéfőző vezérléséhez

Szakdolgozat címe angolul:

Mobile application for smart coffee maker

Intézményi konzulens: Külső konzulens:

Kiss Dániel

Alk.	Dátum	Tartalom	Aláírás
1.	2022. 03. 10.	Tématerv egyeztetése	Kiss Dániel
2.	2022. 03. 31.	Hardveres építőelemek	Kiss Dániel
3.	2022. 04. 21.	Az alkalmazással szembeni elvárások	Kiss Dániel
4.	2022. 05. 09.	Rendszerterv áttekintése	Kiss Dániel

A hallgató a Szakdolgozat I. tantárgy követelményét teljesítette, beszámolóra bocsátható.

Budapest, 2022. május 11.

Kiss Dániel
.....
intézményi konzulens

KONZULTÁCIÓS NAPLÓ

Hallgató neve:	Neptun kód:	Tagozat:
Makai Krisztina Katalin	[redacted]	**nappali**

Telefon: Levelezési cím (pl: lakcím):

[redacted]

Szakdolgozat címe magyarul:

Mobilalkalmazás fejlesztése okos kávéfőző vezérléséhez

Szakdolgozat címe angolul:

Mobile application for smart coffee maker

Intézményi konzulens: Külső konzulens:

Kiss Dániel

Alk.	Dátum	Tartalom	Aláírás
1.	2022. 09. 20.	Megvalósítási terv áttekintése	Kiss Dániel
2.	2022. 10. 10.	Hardveres komponensek tesztelése	Kiss Dániel
3.	2022. 11. 21.	Megvalósítás áttekintése	Kiss Dániel
4.	2022. 12. 06	Tartalmi áttekintés	Kiss Dániel

A hallgató a Szakdolgozat II. tantárgy követelményét teljesítette, védésre bocsátható.

Budapest, 2022. december 13.

Kiss Dániel
.....
intézményi konzulens

TARTALOMJEGYZÉK

1. BEVEZETÉS

A kávé a világ legszélesebb körben fogyasztott itala és az egyik legfontosabb kereskedelmi termék. A 15. század végén az első kávéház megnyitásával jelentősen megnőtt a kávéfogyasztás világszerte [1]. A növekedésének okai között szerepel a minőségi kávéfajta használat, a mezőgazdaság fejlődése, kávéfőző variánsok megjelenése, aminek következtében megjelentek specifikusabb kávézási formák, kultúrák és végül, de nem utolsó sorban a kávéimázs változása. A 2000-es évek végén pedig megjelent az úgynevezett *speciality* kávé mozgalom, melynek célja, hogy a legjobb formában kerüljön a kávé a fogyasztója elé [2].

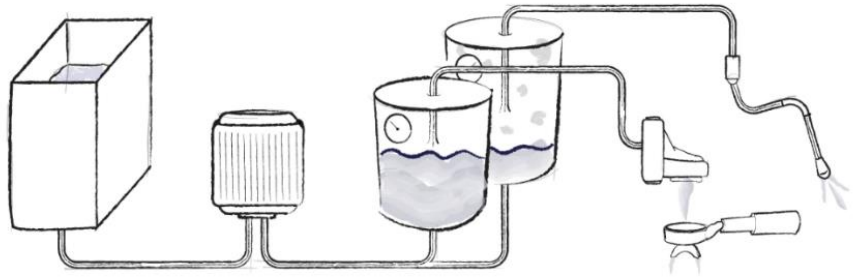
A kávékészítés négy alapelve a tiszta, megfelelő hőmérsékletű (90-96 °C-os) víz, a megfelelő kávé és víz arány, az elkészítési módhoz gondosan megválasztott őrlési finomság és végül a friss kávé [3], [4]. A tökéletes kávéhoz a négy alapelv megfelelő alkalmazása szükséges, de a végeredmény nagyban függ a kávékészítési eljárástól. Ugyanazt a kávét elkészítve különböző eljárásokkal, a kész kávé is más és más lesz [5]. Az eszpresszó kávéfőző gép eljárását úgynevezett túlnyomásos kivonatolásnak nevezzük, amikor forró víz folyik keresztül a finom őrlésű kávén. A végeredményként kapott kávé íze intenzív, de finoman édes.

A dolgozatom célja, hogy egy olyan működő kávéfőzőt és egy hozzá tartozó mobil alkalmazást készítsek, aminek segítségével a felhasználó képes távolról irányítani a kávéfőzőt, illetve létrehozni olyan főzési konfigurációkat, amik segítségével a kávé minősége jobbá tehető. Minden olyan beállításra, paraméterezésre lehetőséget adjon a kellő kávé főzés-tudományi szakértelemmel rendelkező felhasználónak, amivel minőségi kávé készíthető.

1.1. Eszpresszó kávéfőző működése

A mai modern kávéfőzőkben elképesztő technológia rejlik. Azonban a legtöbb gép esetében, amíg a vízből kávé keletkezik, addig a kávéfőző négy jól elkülöníthető fő részen halad keresztül [6]:

- Víztartály
- Pumpa
- Kazán
- Kifolyócsőr



1.1. ábra. Eszpresszó kávéfőző általános felépítése

1.1.1. Víztartály

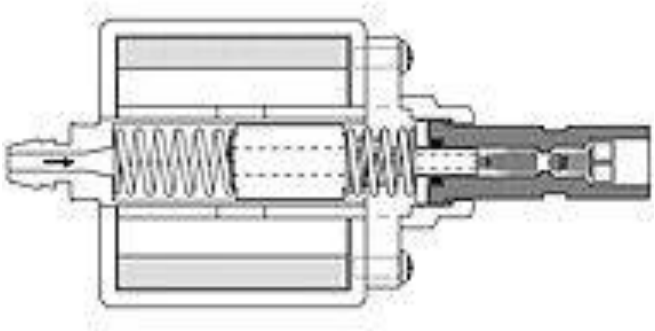
Minden kávéfőzőnek szüksége van vízre, amely két forrásból származhat. Az egyik, a beépített tartály, amely a legtöbb otthoni gépben megtalálható. A másik megoldás, ha a kávéfőző közvetlenül a vízvezetékre van kötve. Ennek nagy előnye, hogy nem kell újra tölteni a tartályt. A csúcskategóriás készülékekben pedig előfordul a kettő egyidejű használata.

1.1.2. Pumpa

Ahhoz, hogy a víz megfelelő erővel haladjon végig a kávéfőzőn, 9±1 bar nyomásra van szükség. Ezt a nyomást a pumpa állítja elő. A modern eszpresszó gépekben két féle pumpát használnak. Elektromágneses elven működő vibrációs pumpát, valamint forgólapátos mechanikai pumpát.

Vibrációs pumpa

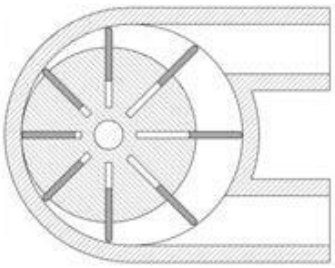
A vibrációs pumpa belsejében egy tekercsben van elhelyezve a mágneshez erősített dugattyú, amely elektromos áram hatására gyorsan mozog előre-hátra, ezáltal nyomva át a vizet a gépen. Egy átlagos vibrációs pumpa nagyjából másodpercenként hatvan nyomást képes elérni.



1.2. ábra. Vibrációs pumpa

Forgólapátos pumpa

A forgólapátos pumpát motor hajtja, amely csúszó lapátokkal tagolt korongot forgat. A lapok a pumpa belsejét kamrákra osztják. Abból kifolyólag, hogy a korong középpontja nem a pumpa közepén helyezkedik el (lásd 2.3 ábra), a forgás során a lapok a pumpa belső falához nyomódnak, csökkentve a kamrák méretét, nyomást hozva létre. A víz a nagy kamrák felől lép be és a zsugorodásukkor kiszorul.



1.3. ábra. Forgólapátos pumpa

Hosszú idő óta komoly vitatémát képez a szakértők körében, hogy melyik pumpa a megfelelő választás a tökéletes eszpresszó kávé főzéséhez, azonban minkét pumpának relatív előnyei vannak. A vibrációs pumpa kisebb, így kevesebb helyet foglal, emellett olcsóbb és könnyebb cserélni meghibásodás esetén. Ezzel szemben a forgólapátos pumpa pedig halkabb, egyenletesebb nyomást képes biztosítani és általában hosszabb az élettartama. Emellett nagyon megoszlanak a vélemények, hogy a kávé ízére és állagára hatással van-e a pumpa működési elve, azonban mindkét pumpával kiváló eszpresszó főzhető [6].

1.1.3. Kazán

A víz hőmérsékletében egy minimális eltérés az ideálistól is befolyásolhatja a kávé minőségét. Az ezért felelős alkatrész a kazán. Eszpresszó kávéfőzők esetében a hőmérsékletet legtöbbször PID-vel szokták szabályozni. Az egyik tipikus PID-szabályozó alkalmazás a hőmérséklet szabályozás. Ekkor egy hőmérsékletmérő szenzor bemenetét használja a szabályozó, és annak kimenete össze van kapcsolva egy vezérlőelemmel, jelen esetben egy fűtőelemmel. Viszonylag egyszerű struktúrája ellenére is igen nagy megbízhatósággal teljesít, ami a kávé minőségének javítása szempontjából nagy előny.

1.1.4. Kifolyócsőr

Ahogy a víz végighalad az eszpresszó gépen, ez a végső állomása. Közvetlenül innen folyik ki a csészébe a kávé.

2. MEGVALÓSÍTÁSI LEHETŐSÉGEK ÁTTEKINTÉSE

2.1. Eszpresszó kávéfőző

A piacon napjainkban rengeteg féle típusú kávéfőző megtalálható, többek között automata kávéfőző gépek, eszpresszó kávéfőzők, kotyogós-, kapszulás-, illetve kávépárnás kávéfőzők stb. A cél az, hogy egy olyan kávéfőzőn dolgozzak, amelynek a felépítése nagyban hasonlít a legtöbb eszpresszó kávéfőző gépekre, ezáltal minimális módosítással át lehessen emelni a megoldásomat bármilyen másik kávéfőzőbe.

A dolgozatom során a prototípus okos kávéfőző alapjául egy Philips Senseo HD7810-et fogok használni. Ez egy kávépárnás eszpresszó kávéfőző gép, az egyik legelterjedtebb az országban, emellett nagyon olcsó és pofon egyszerű a használata.

Ez a kávéfőző a bevezetésben elemzett általános felépítést tekintve, tartalmaz egy víztartályt, egy vibrációs pumpát, egy kazánt és természetesen egy kifolyócsőrt. Ezeket kiegészítve szükség lesz még egy saját vezérlő panelre, amelyhez egy okostelefon, vagy tablet képes lesz kapcsolódni.

2.2. Kapcsolódási lehetőségek

A hálózati protokoll kiválasztása az első lépések között kell legyen egy okos eszköz fejlesztésénél. A legnépszerűbb vezeték nélküli technológiák a Zigbee, Wi-Fi, Bluetooth és a Z-Wave [7].

Zigbee

A Zigbee egy IEEE 802.15.4-es hálózati szabványon alapuló, vezetéknélküli kommunikációs protokoll, amely három frekvencián működik: 868MZh-en, 915MHz-en és 2.4GHz-en. A Wi-Fi és a Bluetooth alternatívája olyan alacsony fogyasztású eszközöknél, amelyek nem igényelnek nagy sávszélességet. A használatához szükség van egy *hub*-ra vagy átjáróra, amelyen az eszköz kommunikál. Úgynevezett *mesh* hálózatban működik, melyben a hálózati csomópontok jeltovábbítóként is funkcionálnak, ami megnöveli a központi egység hatósugarát [8].

Wi-Fi

Napjainkban a legszélesebb körben elterjedt vezetéknélküli technológia a Wi-Fi, amely az IEEE 802.11-es szabványra épül. 2.4GHz-es és 5GHz-es frekvencián működik Az okos otthonokban a Wi-Fi-t többnyire előszeretettel használják az okos készülékek irányítására, monitorozására.

Bluetooth

A Bluetooth egy nagyon alacsony költségű, rövid hatótávú vezeték nélküli technológia, amely az IEEE 802.15.1-es szabványon alapul. Adó-vevő microchipeket használ minden eszközben, ezek segítségével történik a kommunikáció, amely során két eszköz közvetlenül tud egymáshoz kapcsolódni. Nagyon alacsony energiaszükségletű, azonban jóval lassabb adatátviteli sebességre képes, mint a többi vizsgált technológia. Ebből kifolyólag akkor szokták alkalmazni, ha nagyon kis adatmennyiségeket kell küldeni egyik eszközről a másikra.

Z-Wave

A Z-Wave egy új, nagyon alacsony energiafogyasztású vezeték nélküli okos otthon technológia. 900MHz-en kommunikál, ennek köszönhetően szinte nincs interferencia a Z-Wave hálózatokban. A rendszer kiépítése kezdetben többe kerül, viszont, ha túl sok különböző eszköz csatlakozik egy időben a például Wi-Fi-re, akkor a Z-Wave használatával kiküszöbölhető az interferenciából fakadó probléma, mivel nem ugyanazon a hullámhosszon működik, mint a Wi-Fi.

2.3. MCU

ESP8266

Az ESP8266 egy 32 bites mikrokontroller, ami az elmúlt években igen elterjedt a kedvező ára miatt. Rengeteg modul formában kapható kezdve a legegyszerűbbtől, egészen az összetettebb fejlesztő panelekig. A NodeMCU-ESP8266 CP illetve a CH verziója is szóba jöhet a prototípus tervezésekor. A különbség csupán az USB chipekben rejlik [9].

Technikai specifikáció:

- 32-bit RISC mikroprocesszor
- 80MHz-es órajel (160MHz)
- 64 kB + 96 kB RAM
- 1-4 MB Flash memória (külön IC)
- 16 GPIO port
- 1 db analóg bemenet
- IEEE802.11b/g/n Wi-Fi

A technikai specifikáció mellett elengedhetetlen megemlíteni, hogy többféle módon programozható:

- AT parancsok: gyári firmware képes értelmezni az AT parancsokat

- NodeMCU: úgynevezett *lua* scriptekkel futtathatunk programokat a NodeMCU operációs rendszer alatt
- Arduino: C nyelven írhatunk rá programokat, köszönhetően annak, hogy az Arduino környezet támogatja az ESP modulokat és fejlesztő paneleket

ESP32

Az ESP32 nagy előnye, hogy jóval több portja van, mint az ESP8266-nak. Szinte bármelyik kialakítás alkalmas lehet a megvalósításhoz.

Technikai specifikáció:

- Xtensa® single-/dual-core 32-bit LX6 processzor
- 80MHz-es órajel (240MHz)
- 520 kB SRAM
- 4 MB Flash memória
- 34 GPIO port (16 PWM)
- Bluetooth 4.2/BLE
- IEEE802.11b/g/n Wi-Fi

A mikrokontroller programozását tekintve, az ESP8266 kapcsán említettek mellett még számos szoftver, fejlesztőkörnyezet támogatja az ESP32-t, ezek közül néhányat emelnék ki:

- ESP-IDF (Espressif IoT Development Framework): a hivatalos fejlesztői környezete az ESP32-nek
- MicroPython: Python 3 programozási nyelv implementációja
- Mongoose OS: C és JavaScript nyelven programozhatunk

ESP32-S2

Az ESP32-höz képest a legnagyobb előnye a modernebb architektúrával rendelkező Xtensa® LX7-es processzor. Azonban ez nem rendelkezik integrált Bluetooth modullal.

Technikai specifikáció:

- Xtensa® single-core 32-bit LX7 processzor
- 80MHz-es órajel (240MHz)
- 320 Kb SRAM
- 2-4 MB Flash memória
- 43 GPIO port (8 PWM)
- IEEE 802.11 b/g/n Wi-Fi

A programozását tekintve szinte teljesen megegyezik az ESP32-vel.

ATMega328

Az egyik legelterjedtebb fejlesztőpanel az Arduino Uno. Ez egy ATMega328P típusú mikrokontrollerrel ellátott fejlesztőpanel. Az ATmega328P nem rendelkezik integrált Wi-Fi modullal, viszont ennek kiküszöbölésére az Arduino Uno Wi-Fi fejlesztőpanelre integráltak egy ESP8266Wi-Fi modult is.

Technikai specifikáció:

- 16MHz-es órajel
- 32 kB Flash memória
- 2 kB SRAM
- 1 kB EEPROM
- 6 analóg input port
- 14 digitális GPIO port (6 PWM)

Az alábbi szoftverek teszik lehetővé a mikrokontroller programozását online és offline környezetben egyaránt:

- Arduino IDE: nyílt forráskódú IDE, C nyelven írhatunk programot rá
- Arduino CLI: parancssorról lehet programozni, az Arduino IDE és a Web Editor erre épül
- Web Editor: böngészőből lehet programot írni vármely Arduinora

2.4. ULKA Vibrációs pumpa

A kávéfőző, amivel dolgozni fogok adott. Ez a kávéfőző rendelkezik egy vibrációs pumpával, amelyről, a forgólapátos pumpával együtt, megoszlanak a vélemények még a kávészakértők körében is, viszont a dolgozatomhoz tökéletesen megfelel. Mivel a vibrációs pumpa egy mágneses mező által működtetett dugattyú, így a vízáramlás mértékét a szivattyúhoz küldött impulzusok frekvenciája határozza meg.

Áramlási sebesség

Egy eszpresszó kávéfőző gép áramlási sebessége (*flow rate*) az a mértékegység, mellyel azt határozzuk meg, hogy mennyi víz halad keresztül a kávéfőzőn, amíg a pumpa aktív. A legtöbb kereskedelmi kávéfőző áramlási sebessége 250 és 500 g/30s közé tehető, azonban az ideális intervallum 200 és 280 g/30s közé tehető.

A legtöbb pumpa három vezetékkel és egy DC motorral rendelkezik. Ebből két vezeték felel a motor működéséért, a harmadik pedig egy 5V-os vezeték a PWM-es sebesség

szabályozáshoz. Néhány pumpa esetén csak két vezeték található, ez azt jelenti, hogy nincsen külön vezeték a pumpa sebességének szabályozásához, azonban ebben az esetben is lehetőség van a motort PWM-mel szabályozni [10].

A dolgozatom alapjául szolgáló kávéfőzőhöz tartozó pumpa az alábbi specifikációkkal rendelkezik [11]:

ULKA Vibrációs pumpa HF 22W 230V 50Hz (1331021)

- tömlővég illesztés ø6.6 mm -ø6.6 mm
- max. nyomás 2,6 bar
- max. víz hőmérséklet 20°C
- integrált dióda (1N 4007P)
- 2 db vezeték csatlakozó 4.7x0.8 mm

Mint ahogy korábban említettem, ahhoz, hogy az eszpresszó kávéhoz a víz megfelelő erővel haladjon végig a kávéfőzőn, 9±1 bar nyomásra van szükség. Ennek előállítására a fenti pumpa a specifikációja alapján nem alkalmas. Ahhoz, hogy a kívánt nyomást elérjük, másik, magasabb teljesítményű pumpát kellene használni. Azonban a prototípus fejlesztéséhez a költséghatékonyság miatt megelégszem a már meglévő pumpával. A későbbiekben pedig meg lesz a lehetőségem olyan vibrációs pumpát találni, amely képes az ideális nyomást előállítani és kompatibilis a többi hardverrel. Az ULKA-nak több vibrációs pumpája is megfelelne a célnak, többek között az EX7GW, az EX5GW, az EX5, az EX4, az EP8GW, az EP5GW, az EP5GW, vagy az EP5 [11].

2.5. Philips Senseo fűtőelem

A kávéfőzőbe eredetileg beszerelt kazánt fogom használni a saját okos kávéfőző elkészítéséhez. A fűtőelemet egy hőmérséklet szenzor segítségével PID-vel lehet szabályozni. Hogyha tudjuk mérni a hőmérsékletet, akkor egy relé segítségével tudjuk be-, illetve kikapcsolni a fűtőelemet. A leolvasott hőmérsékletből és a célhőmérsékletből kiszámoljuk a hibát, összeadjuk a PID értékeket és PWM jelet küldünk a relére [12].

2.6. Mobil platform

Ha natív mobil alkalmazás fejlesztésről beszélünk, akkor két piacvezető platform közül választhatunk. Ezek az Android és az iOS. Mindkét platform hatalmas lehetőségeket nyújt a fejlesztés terén, de mielőtt kiválasztom a dolgozat szempontjából ideális platformot, hasonlítsuk össze őket.

Operációs rendszer

Az iOS fejlesztéshez Mac OS X 10.6 operációs rendszert futtató Macintosh számítógépre van szükség [13]. Mivel az alkalmazások relatív kis méretűek, és nincs nagy processzor igényük, így nem szükséges különösen erős gép a fejlesztéshez.

Ezzel szemben Android fejlesztés nincs ennyire lekorlátozva. A fejlesztés történhet szinte bármilyen operációs rendszeren, Windows-on (XP, vagy újabb), Mac OS X-en (10.5.8 vagy újabb), és Linuxon is (kernel 2.6 vagy újabb).

IDE

Az Apple integrált fejlesztő környezete az XCode, ami olyan eszközöket tartalmaz, amelyek segítségével az egész fejlesztési folyamat lefedhető az alkalmazás létrehozásától az App Store-ba kerülésig. A másik sokak által kedvelt fejlesztőkörnyezet, a Microsoft Xamarin alkalmazása. Ez egy nyílt forráskódú, ingyenes *cross-platform* környezet, Android és iOS alapú alkalmazás fejlesztéshez .NET és C# használatával. Könnyebb a használata, mint az XCode-é, főleg, ha valaki nincs Mac OS-hez szokva.

Az Android-alkalmazások fejlesztéséhez több fejlesztőkörnyezet közül lehet választani. Az egyik népszerű környezet a Visual Studio, ami integrálódott a Xamarinnal, így C# nyelven, .NET keretrendszer használatával lehet mobil applikációt fejleszteni egyaránt Androidra és iOS-re is. Egy másik lehetőség az Eclipse használata, ugyanis használható Android alkalmazás fejlesztésére is. Több különböző nyelvet támogat, mint a C, C++, C#, PHP, Perl és a Ruby, de elsődlegesen a Javat. Végül, de nem utolsó sorban a Google hivatalos Android fejlesztésre szánt fejlesztőkörnyezete, az Android Studio, amely különböző funkciókkal rendelkezik, megkönnyítve az alkalmazásfejlesztést.

3. RÉSZLETES SPECIFIKÁCIÓ

A teljes megvalósítási terv elkészítéséhez szükséges definiálni, hogy milyen részfolyamatokra, komponensekre lesz szükség. Az egyik a hardveres rész az MCU oldaláról, a másik pedig a szoftveres része, ami maga a mobil alkalmazás és annak specifikációja.

3.1. Választott konfiguráció és indoklása

3.1.1. Kritériumok

Ahhoz, hogy az előző fejezetben elemzett lehetőségek közül ki tudjam választani a megfelelő megoldási módszereket elhanyagolhatatlan, hogy felállítsak egy kritériumrendszert, amely segítséget fog nyújtani a továbbiakban. Ezek a felállított kritériumok az alábbiak:

- **Költséghatékonyság**
  A dolgozat megvalósítása során talán a legnagyobb korlátozó tényező az ár. Mindamellett, hogy nem célom egy csúcskategóriás terméket előállítani, meg kell szabnom egy költséghatárt. Nem csupán azért, mert véges kereteken belül dolgozhatok, hanem hogy a kidolgozott prototípus után esetlegesen érdeklődőket ne tántorítsa el az anyagi oldal.
- **Használhatóság**
  A felhasználói elvárásoknak eleget téve, szeretnék egy olyan alkalmazást létrehozni, amely letisztult, minimalista és könnyen használható.
- **Modern technológiák alkalmazása**
  Olyan eszközöket és technológiákat szeretnék alkalmazni, amelyek ötvözik a rendelkezésre álló lehető legmodernebb és leginnovatívabb technológiát.

3.1.2. Működési elvek

A fent felsoroltak mellett technikai megszorításokat is alkalmazok:

- **Kompatibilitás**
  A dolgozatom egyik legfőbb célja, hogy egy olyan megoldást dolgozzak ki, amely minimális módosításokkal átültethető komolyabb kávéfőző gépekbe.
- **Bővíthetőség**
  Mivel talán a legegyszerűbb kávéfőző típussal készítem a prototípust, ezért fontos szempont számomra a kompatibilitással karöltve, hogy nem csak átültethető legyen, hanem bővíthető is legyen, például egy gőzkarral és annak vezérlésével a megoldásom.

- **Biztonság**
  Nagyon természetesnek tűnő, de annál fontosabb szempont, hogy vízzel és nyomással dolgozni nem teljesen veszélytelen feladat. Viszont természetesen a felhasználó számára is az okos kávéfőzőnek teljes mértékben biztonságosnak kell lennie.
- **Megbízható vezetéknélküli kapcsolat**
  Minden okos eszköznek, így egy okos kávéfőzőnek is elengedhetetlen feltétele, hogy képes legyen gyors és megbízható vezeték nélküli kapcsolat kialakítására.

A kritériumok és működési elvek megfogalmazása után a következő feladat, hogy kiválasszam a használni kívánt komponenseket.

3.1.3. Wi-Fi kapcsolódás

A mikrokontroller kiválasztásához az elsődleges szempont a vezeték nélküli kapcsolódási technológia. A korábban megismert négy technológia közül fogok választani egyet, amelyik a lehető legjobban eleget tesz a felsorolt kritériumrendszernek. Az alábbi táblázat összefoglalja a tárgyalt vezeték nélküli technológiák-, a dolgozatom szempontjából a legfontosabb tulajdonságait összehasonlítva.

	ZigBee	**Z-Wave**	**Wi-Fi**	**Bluetooth**
Fogyasztás	100 mw	1 mw	High	10 mw
Hatótávolság	100 m	30 m	1000 m	10 m
Költség	Alacsony	Magas	Közepes	Nagyon alacsony
Méretezhetőség	6000	> 6000	32	20
Interopera-bilitás	Ugyanazon gyártó	Különböző gyártók	Wi-Fi kompatibilis eszközök	Bluetooth kompatibilis eszközök

3.1. táblázat. Vezeték nélküli technológiák összehasonlítása

A ZigBee-t és a Z-Wave-et jellemzően akkor használják, amikor egy sok eszközös nagyobb hálózatot építenek ki [8]. A legnagyobb különbség a két technológia között, hogy különböző rádiófrekvencián kommunikálnak, ebből adódik, hogy a Z-Wave-nek nagyobb a hatótávolsága, míg a ZigBee több adatot képes közvetíteni [14]. Bár nagyon csábítóan hangzik egy viszonylag új technológia használata, mivel a dolgozatom esetében nem beszélhetünk sok eszközről, így plusz költségekkel járna bármely technológia használata a kettő közül. Tehát más megoldást kell választanom a vezeték nélküli kapcsolódásra.

A maradék két versenyben lévő technológia a Bluetooth és a Wi-Fi. Először is egy nagyon sarkalatos különbséget emelnék ki, ami nem más, mint a biztonság. Ahhoz, hogy például a telefonommal kapcsolódjak Bluetooth-on keresztül a vezeték nélküli fülhallgatómhoz, elég egy bizonyos távolságon belül lennie a két eszköznek és jelszó nélkül létrejön a kapcsolat. Ezzel szemben Wi-Fi esetén használhatók bizonyos biztonsági protokollok, mint a WEP, WPA, WPA2 és WPA3. A Wi-Fi mellett szól még, hogy minimális költség többletért cserébe jóval gyorsabb adatátvitelre képes, mint a Bluetooth, illetve sokkal nagyobb a hatótávolsága is. Minezeket egybe véve a vezeték nélküli kapcsolat kialakítására a Wi-fi használata a legmegfelelőbb számomra.

3.1.4. Mikrokontroller

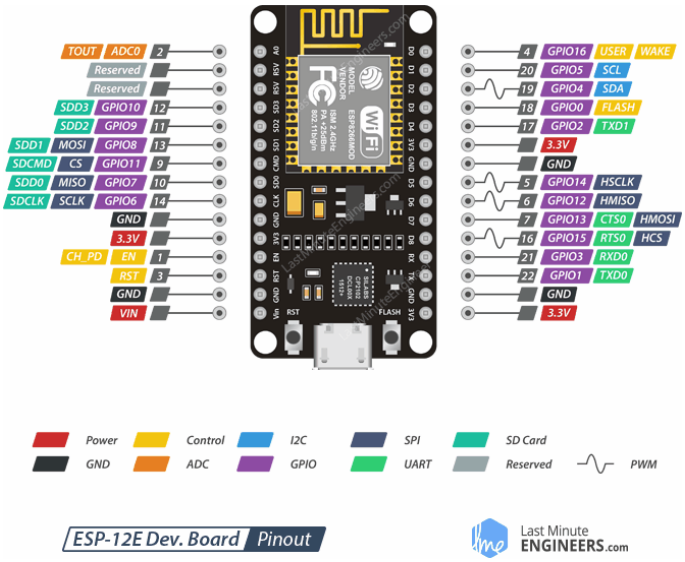
Ahogy korábban már említettem, a felhasznált kávéfőző mellett szükség lesz egy fejlesztő panelre, amely rendelkezik beépített Wi-Fi modullal. A mikrokontroller felé támasztott igény alapján az alábbiak elérhetőek és felelnek meg az elvárásoknak:

- ESP8266 modullal rendelkező fejlesztőpanelek
- ESP32-vel rendelkező fejlesztőpanelek
- ESP32-S2-vel rendelkező fejlesztőpanelek

	ESP8266	ESP32	ESP32-S2	ATMega328P
Processzor	32-bit RISC processzor	Xtensa® single-/dual-core 32-bit LX6 processzor	Xtensa® single-core 32-bit LX7 processzor	8-bit RISC processzor 28P
Órajel	160MHz-ig	240MHz-ig	240MHz-ig	16MHz
Wi-Fi	IEEE 802.11b/g/n	IEEE 802.11b/g/n	IEEE 802.11b/g/n	Integrált ESP8266Wi-Fi
GPIO pinek	16	34	43	14
PWM	16	16	8	6

3.2. táblázat. Mikrokontrollerek összehasonlítása

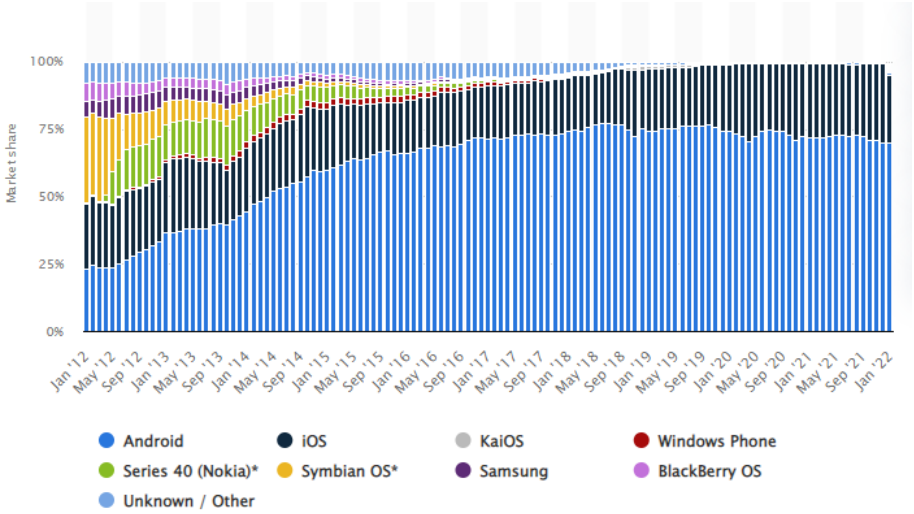
Ezek közül mind a három teljes mértékben kielégíti az MCU felé támasztott igényeimet, azonban a tanulmányaim során megismerkedtem többek között az Arduino és az ESP8266 programozásával is, használatuk egyszerű, rengeteg könyvtár és fórum bejegyzés érhető el hozzájuk, megkönnyítve a fejlesztést és a felmerülő problémák megoldását. Azonban az Arduino UNO Wi-Fi fejlesztőpanel meglepően drága és nem szerezhető be könnyen, szemben az ESP8266 NodeMcu-val, amiből korábbi egyetemi órák alkalmával bevásároltam. Ezért a választásom az ESP8266 NodeMcu -ra esik. Az ESP32/ESP32-S2 is tökéletesen megfelelne a kivitelezéshez, azonban erőforrás pazarlásnak érezném, ha ezekkel felszerelt panelt választanék. Azonban, ha a későbbiekben a NodeMcu használata közben olyan nem várt problémába ütköznék, amely a mikrokontroller lecserélését igényelné, így van bőven opcióm ezek közül választani.



3.1. ábra. Az ESP8266 NodeMcu részletes pinout diagramja (Forrás: [18])

3.1.5. Android platform

Mielőtt bárki mobil alkalmazás fejlesztésbe kezd, az elsődleges döntést meg kell hozni, hogy milyen mobil platformra fog történni a fejlesztés. Mára már két vezető platform nőtte ki magát, az iOS és az Android. Felmerült annak az ötlete is, hogy mindkét platformra készüljön el az alkalmazás, amelyhez elérhető olyan fejlesztőeszköz, amely *cross-platform* fejlesztést is támogat. A legelterjedtebb ilyen eszköz a Xamarin, amellyel C# nyelven .NET keretrendszert használva lehet iOS-re és Androidra is egyidejűleg alkalmazást írni. Azonban a Xamarinnak gyengébb a dokumentáltsága, mint a platform függő fejlesztőeszközöknek, kisebb is a felhasználói bázisa, illetve kevesebb szoftverkönyvtár érhető el.



3.2. ábra. Mobil operációs rendszerek eloszlása az elmúlt tíz évben (Forrás: [15])

Tehát a választást a két fő platform közül kell meghozni. Az elmúlt néhány évben, lassan, de egyre nagyobb teret hódít magának az iOS, azonban a statisztikákból egyértelműen kiderül, hogy még mindig messze az Android operációs rendszert használók vannak többségben. 2022-es felmérés szerint a felhasználók kb. 70%-a választja inkább az Androidos készülékeket [15]. Annak ellenére, hogy én a kisebbség bázisát erősítem, úgy döntöttem, hogy a megfelelő választás az, ha minél több embert el tud érni egy alkalmazás, így az Android platformot választottam a mobilalkalmazás fejlesztéséhez.

Az Android hivatalos programozási nyelve a Kotlin, azonban sok más alternatív programnyelv áll rendelkezésre, mint a Java, C++, C# és még jó néhány. Korábban a Java volt az Android hivatalos programozási nyelve, azonban 2017 óta az új Kotlin nyelv vette át a helyét. A Java a mai napig egy nagyon népszerű programozási nyelv, rengetegen használják szerte a világon. Nagyon sok oktatóanyag, könyv, kurzus elérhető ingyen, azonban ez a Kotlinról nem mondható el [16]. Ellenben fejlesztői szemmel nézve több kiemelkedő előnye van a Kotlinnak:

- **Kiforrott nyelv és környezet:** Más programnyelvektől eltérően számos szakaszon ment keresztül a végleges 1.0-ás verzió kiadásig. Ez azt jelenti, hogy számos nyelvi probléma megoldásra került a végleges verzióig.
- **Könnyebb Android fejlesztés:** A Kotlin egy modern programnyelv, amely segítségével a fejlesztő könnyebben írhat jobb Androidos alkalmazást.
- **Bugok és hibák minimalizálása:** A Kotlin fordító az ún. *fail-fast* -ra törekszik, ami annyit teszt, hogy azonnal jelzi, hogyha bármi olyat talál a kódban, ami hibát indukálhat. Ennek köszönhetően csökken a futás idejű hibák száma, illetve a javításba vetett energia befektetés.
- **Biztonságosabb:** A gyakori tervezési hibák elkerülhetők a Kotlin használatával, amely így kevesebb rendszerhibát eredményez.
- **Tömörebb:** Sok esetben jóval rövidebb kóddal megoldható egy-egy probléma, mint bármely másik programozási nyelvvel, így növelheti a produktivitást. Egy Javaban írt 50 soros osztály megírható néhány sorban Kotlin nyelven.

A felsorolt előnyei miatt, illetve a szakmai kíváncsiságomból kifolyólag a mobil applikációt az Android hivatalos programozási nyelvén, Kotlinban fogom írni.

	iOS	Android
Operációs rendszer	Mac OS	Windows, Linux, Mac OS
IDE	Xcode, Xamarin	Android Studio, Xamarin, Eclipse
Programozási nyelv	Objective-C, C#	Java, C#, Kotlin
Weboldal	http://developer.apple.com/iphone	http://developer.android.com/

3.3. táblázat. iOS és Android összehasonlítása [13]

Ha már a választott programozási nyelv, az Android hivatalos nyelve, akkor magától értetődő, hogy a hivatalos fejlesztő környezetét használjam, ami nem más, mint az Android Studio IDE. Természetesen támogatja a Kotlint, és nagy előnye, hogy vizuálisan látható az alkalmazás, hogy hogyan fog kinézni mobil telefonon, szimulálni képes a működését és ki lehet próbálni azt. Android Studiot használva Kotlin nyelven írt mobil alkalmazás fejlesztéséhez elengedhetetlen az Android SDK használata. Ez szintén az Android hivatalos fejlesztő készlete, amelyhez rengeteg dokumentáció érhető el, megkönnyítve a fejlesztést.

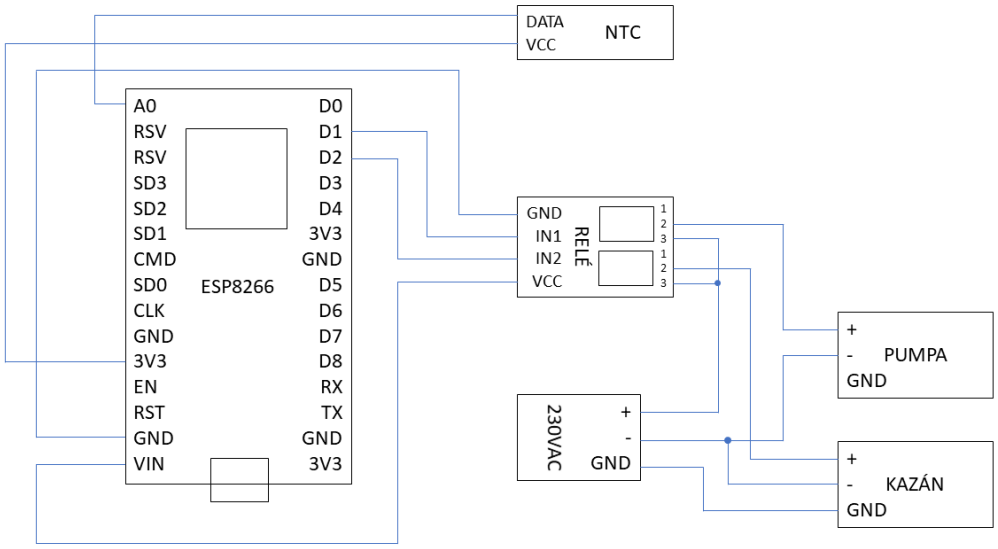
3.2. MCU oldaláról

A prototípus képes lesz a mikrokontrolleren keresztül szabályozni a pumpát és a kazánt. Ez lesz a mikrokontroller két fő feladata. A pumpa és a kazán szabályozásához szükség lesz egy-egy relére. Ezeket megfelelően bekötve fogom tudni irányítani a kávéfőzőn áthaladó víz mennyiségét, illetve sebességét. A víz hőmérsékletének szabályozásához szükség lesz egy hőmérséklet mérő szenzorra, hogy azt PID-vel lehessen szabályozni a kazánt.

Az alábbi GPIO pinoutokra lesz szükség:

- 1 db – IN hőmérséklet mérő szenzor
- 1 db – OUT kazán szabályozás
- 1 db – OUT pumpa szabályozás

3.2.1. Kapcsolási rajz



3.3. ábra. Kávéfőző kapcsolási rajza

3.3. Szoftver specifikáció

A fent említett hardverek vezérlését egy mobil applikációról, vezeték nélküli kapcsolaton keresztül fogja tudni a felhasználó elvégezni. Különböző beállítási lehetőségek segítségével lehet majd specifikálni a főzési folyamatot, ezért fontos, hogy bizonyos kritériumoknak eleget tegyen a mobil alkalmazás.

Ezek a kritériumok az alkalmazással kapcsolatban a következők:

- letisztult, felhasználóbarát felület
- könnyű kezelhetőség
- egyértelműség
- használhatóság
- reszponzivitás
- szép dizájn

A felhasználó az alábbi tervezett funkciókat lesz képes elérni és használni az applikáción keresztül:

- főzési ciklus meghatározása
- átfolyási ráta meghatározása
- főzési hőmérséklet meghatározása
- főzési idő meghatározása

- főzés indítása
- beállított főzési konfiguráció mentése
- korábban elmentett főzési konfigurációk listázása

Az alkalmazás megnyitása után a felhasználót a fő képernyő fogadja majd, ahol listázva lesznek az alapértelmezett főzési konfigurációk. Ehhez a listához lesz képes hozzáadni a felhasználó újakat, módosítani, illetve törölni.

Megnyitva egy főzési konfigurációt, elérhetővé fog válni az összes beállított érték, itt lesz lehetőség a módosításra, illetve törlésre. A fő menüből lehet majd új főzési konfigurációt létrehozni. A felhasználó beállítva a főzési folyamat minden paraméterét, képes lesz elmenteni, illetve el is indítani a főzést a telefonjáról.

4. TERVEZÉS ÉS MEGVALÓSÍTÁS

A mobil applikációval kávéfőzési folyamatot szeretnék felhasználó által konfigurálhatóan megvalósítani. Egy kávéfőzési folyamat tetszőleges számú lépésből állhat, és ezeket a lépéseket lehet majd konfigurálni. A felhasználó egy lépésen belül három tényezőt állíthat:

- Hőmérsékletet
- Átfolyási rátát
- Időtartamot

Ez a három tényező egyértelműen meghatározza a kávéfőzési folyamat egy lépését. Ilyen lépések szekvenciális sorozata pedig meghatároz egy kávéfőzési folyamatot.

4.1. Biztonság

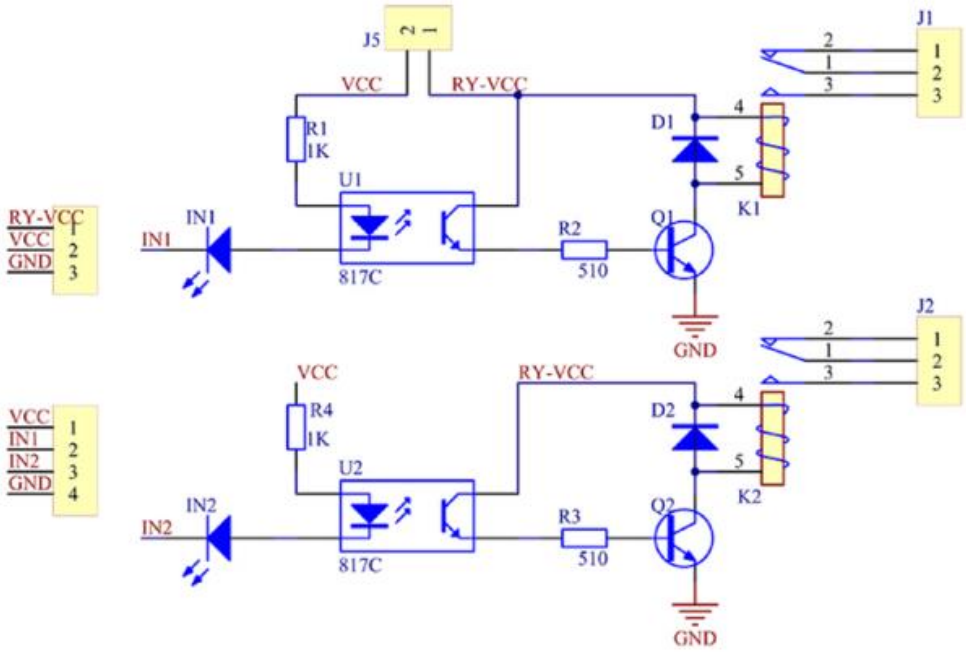
Mindenekelőtt a nagyon fontos szót ejteni a lehetséges veszélyekről és biztonságról, mind a fejlesztői, mind a felhasználói oldalról. A legfontosabb, amire egy kávéfőző esetében oda kell figyelni, az a kazánban a melegítés hatására keletkező nyomás. Éppen ezért hardveresen és szoftveresen is implementáltam biztonsági funkciókat a baleset elkerülése érdekében. Hardver oldalon egy biztonsági szelep gondoskodik a túlnyomásról, a másik oldalon, pedig egy konstans felsőkorlátot határoztam meg, amit, ha az aktuális hőmérséklet elér, akkor a kazán kikapcsol. Ezt az értéket, a biztonság kedvéért úgy választottam meg, hogy belekalkuláltam az NTC termisztor, illetve referencia hőmérő pontatlanságát, így a választott felsőkorlát 95 °C lett.

Elsősorban fejlesztői oldalon, de mindenképp potenciális veszélyforrás magas feszültséggel dolgozni. Természetesen mindent leszigeteltem, azonban rákötöttem a relé kazán oldalára egy próbalámpát, ami akkor világít, ha zárva van az áramkör. A pumpa esetén hallani, hogy mikor van bekapcsolva, viszont a kazán esetén nem, így tehát biztonságosabban folyt a munka.

4.2. A hardver oldali komponensek elkészítése

4.2.1. Feszültség szabályozó

A pumpa és a kazán bemeneti feszültsége 230VAC, valamint ezt a feszültséget szabályozva lehet a pumpa sebességét, illetve a kazán hőmérsékletét kontrollálni. Mivel a mikrokontroller 3,3V, illetve 5V feszültséget képes leadni, ezért szükség lesz egy relé alkalmazására, amelyet a mikrokontrollerrel tudok kapcsolni. Erre tökéletes szolgálatot tesz a BN 2268118, 5V 2-Channel Relé modul. Ez egy kétcsatornás relé modul, amivel mindkét elemet szabályozni tudom. A maximális kimenete: DC 30V / 10A, AC 250V / 10A.



4.1. ábra. BN 2268118, 5V 2-Channel Relé modul kapcsolási rajza (Forrás: [17])

Az alábbiak szerint kell kábelezni a pineket:

- VCC: relé feszültség (5V/DC)
- GND: föld
- IN1: vezérlő jel relé 1 (5V/DC)
- IN2: vezérlő jel relé 2 (5V/DC)

4.2.2. Pumpa kapcsolása

A pumpát ki-, illetve bekapcsolni tudom. Ahhoz, hogy adott vízmennyiséget tudjak pumpálni időegység alatt, szükség van PWM jelre, amellyel a relét egy működési ciklus alatt bizonyos ideig zárt, illetve nyitott állapotba tudom kapcsolni. Azonban a relé szintén csak nyitott, illetve zárt állapotban lehet, digitális jelet képes fogadni, ezért a PWM-et magam valósítottam meg. Tehát a saját megvalósításom százalékos érték alapján képes felosztani a működési ciklust az adott arányban. Például 1 másodperces működési ciklus esetén 75%-os rátával 75 ms-ig zárom a relét, tehát be van kapcsolva, 25 ms-ig pedig nyitva van a relé, tehát ki van kapcsolva a pumpa. A pumpát 100%-os átfolyási rátával teszteltem, egymás után öt tesztet végeztem, és megmértem az átfolyt víz mennyiségét.

Átfolyási ráta (%)	Mennyiség (g)
100%	295 g
100%	279 g
100%	264 g
100%	265 g
100%	271 g

4.1. táblázat. Pumpa teljesítménye 100%-os átfolyási rátával

A következő lépés a megfelelő működési ciklus meghatározása. Ehhez teszteltem a pumpa teljesítményét azonos idővel különböző működési ciklusokkal. Mivel megfigyeltem, hogy a pumpa melegszik és a teljesítménye csökken az egymás utáni gyakori használattól, ezért a teszteseteket random sorrendben végeztem, hogy kiküszöböljem a hibát. A tesztelés eredménye az alábbi táblázatban látható.

	Átfolyási ráta			
Működési ciklus	**25%**	**50%**	**75%**	**95%**
100 ms	100 g	165 g	241 g	283 g
	90 g	168 g	240 g	275 g
	147 g	158 g	244 g	279 g
250 ms	82 g	161 g	225 g	268 g
	76 g	148 g	210 g	275 g
	104 g	153 g	220 g	274 g
500 ms	75 g	151 g	211 g	284 g
	69 g	147 g	213 g	277 g
	75 g	146 g	205 g	246 g
1000 ms	76 g	144 g	207 g	266 g
	73 g	137 g	209 g	267 g
	73 g	144 g	207 g	256 g
2000 ms	72 g	146 g	207 g	277 g
	71 g	138 g	206 g	251 g
	70 g	142 g	208 g	269 g

4.2. táblázat. Pumpa teljesítménye különböző működési ciklusokkal

Ahhoz, hogy ki tudjam választani a megfelelő működési ciklust, meg kell határozzak a 100%-os átfolyási rátához egy referencia értéket. Ehhez a 4.1-es táblázat értékeit átlagoltam, ennek értéke ~275 g víz, 30 másodperc alatt 100%-os átfolyási rátával.

Majd a 4.2-es táblázatban szereplő értékeket működési ráta szerint szintén átlagoltam, összevetettem a referencia értékkel arányosan és kiválasztottam azt a működési ciklust, amely a legjobban ráilleszthető. Így végül a választott működési ciklus az 1000 ms lett.

	25%	50%	75%	95%
100 ms	112 g	164 g	242 g	279 g
250 ms	87 g	154 g	218 g	272 g
500 ms	73 g	148 g	210 g	269 g
1000 ms	**74 g**	**142 g**	**208 g**	**263 g**
2000 ms	71 g	142 g	207 g	266 g

4.3. táblázat. Pumpa teljesítménye átlagolva különböző működési ciklusokkal

4.2.3. Kazán kapcsolása

A kazán esetében hatalmas nehezítő tényező, hogy nem ismert a gyártó, így az adatlapja sem. Mivel a kazánt PID-vel szeretném szabályozni, ezért szükséges ismernem a kazánban lévő víz hőmérsékletét. A kazánba be van építve egy NTC termisztor, ami egy a hőre történő ellenállás értékének változásán alapuló hőmérsékletmérő szenzor. Minél nagyobb a hőmérséklet, az ellenállás értéke annál kisebb, viszont az ellenállás értékének változása nem lineáris. Ahhoz, hogy az ellenállás értékeket hőmérsékletté tudjam alakítani, szükség van a termisztor ellenállására, az anyagállandóra, illetve a Steinhart-Hart formulához szükséges konstansokra, amiket a gyártó, vagy eladó az adatlapján szokott feltüntetni. Mivel nem ismert számomra az adatlap, ezért kísérletezéssel fogok egy hőmérséklet-ellenállás diagramot készíteni, amely segítségével később tudok dolgozni.

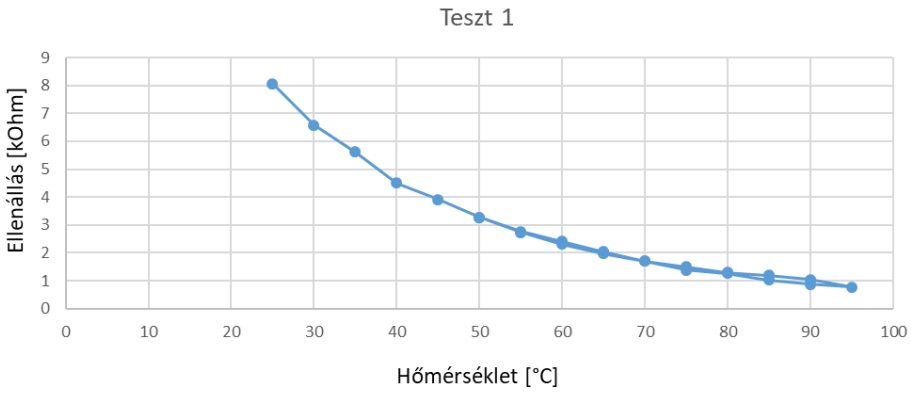
NTC Termisztor tesztelése

A tesztelés során egy konyhai edényt ujjnyira öntöttem fel vízzel, és villanytűzhelyen elkezdtem melegíteni. Egy digitális hőmérőt és az NTC szenzort a vízbe helyezve mértem melegítés közben adott hőmérsékleten a szenzoron az ellenállást és feljegyeztem a 4.4-es táblázatba.

Hőmérséklet (°C)	NTC ellenállás (kΩ)
25	8,06
30	6,58
35	5,62
40	4,5
45	3,91
50	3,27
55	2,77
60	2,41
65	2,04
70	1,71
75	1,38
80	1,27
85	1,02
90	0,87
95	0,77
90	1,04
85	1,2
80	1,3
75	1,5
70	1,71
65	1,97
60	2,31
55	2,73
50	3,27

4.4. táblázat. NTC szenzor tesztelési adatai

Az összegyűjtött adatokat az alábbi 4.2-es diagramon ábrázoltam. Jól látható rajta a görbe jellege, illetve az, hogy a mért hőmérsékletek között nagyságrendileg hogyan változik az ellenállás értéke. Látható, hogy 80 °C és 90 °C között ketté válik a görbe, mert a hiszterézisnek köszönhetően a hőmérséklet növekedés közben más ellenállás értékeket mértem, mint hűlés közben. Ez azt jelenti, hogy mikor 1 kΩ-ot mérek, akkor a hőmérséklet 85 °C és 90 °C között bárhol lehet.



4.2. ábra. NTC szenzor hőmérséklet-ellenállás diagramja

A következő feladat meghatározni a hőmérsékletet az ellenállás alapján a korábban említett Steinhart-Hart formulával. Mivel a hozzá tartozó A, B és C konstansokat szintén a gyártó szokta megadni, így ezeket nekem kell meghatározni a mért ellenállás értékek alapján. Az alábbi képletet használva egy három ismeretlenes egyenletrendszer írható fel, amit megoldva eredményül kapjuk az A, B és C konstansokat.

$$T^oK = \frac{1}{a_0 + a_1 \ln(R_T) + a_2[\ln(R_T)]^3}$$

$$T_1 = \frac{1}{a_0 + a_1 \ln(R_1) + a_2[\ln(R_1)]^3}$$

$$T_2 = \frac{1}{a_0 + a_1 \ln(R_2) + a_2[\ln(R_2)]^3}$$

$$T_3 = \frac{1}{a_0 + a_1 \ln(R_3) + a_2[\ln(R_3)]^3}$$

$$\frac{1}{368{,}15} = A + B \cdot 6{,}6464 + C \cdot 293{,}601$$

$$\frac{1}{348{,}15} = A + B \cdot 7{,}2793 + C \cdot 385{,}72$$

$$\frac{1}{323{,}15} = A + B \cdot 8{,}0925 + C \cdot 529{,}975$$

Az egyenletrendszert megoldva a konstans értékek:

A = 1,64279e-3
B = 1,24461e-4
C = 8,38797e-7

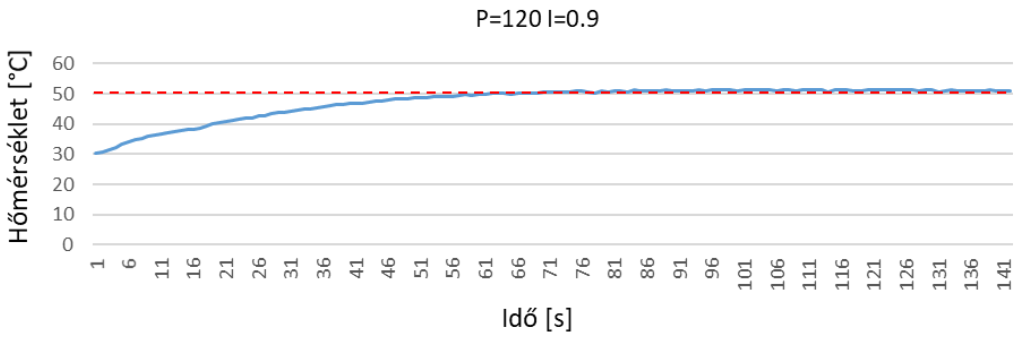
A konstansok ismeretében, pedig végül a Steinhart-Hart formula segítségével felírható a mért hőmérséklet Kelvinben.

$$T = \frac{1}{A + B[\ln(R)] + C[\ln(R)]^3}$$

Ezt az értéket Celsius fokká konvertálva, használatra kész a termisztor. A kazán szintén 230 VAC-vel működik, ehhez is a relét fogom használni, így ki, illetve bekapcsolni tudom. Viszont ahhoz, hogy a megfelelő hőmérsékletre tudja melegíteni a vizet, a szabályozó implementálásához és hangolásához figyelembe kell venni néhány fontos szempontot.

- Először is, fontos, hogy ne legyen túllövés, hiszen majdnem forrásig kell melegíteni a vizet és a túl nagy nyomás következtében esetleg felrobbanhat a kávéfőző.
- Másodszor a lehető legpontosabban be kell álljon a víz hőmérséklete az elvártra, viszont az ehhez kellő idő a prototípust tekintve teljesen elhanyagolható, hiszen a biztonság sokkal fontosabb szempont.

Tehát ezek fényében egy PI szabályozót implementáltam, ami a pumpa során már használt PWM jelet generálva az aktuális hőmérséklet és a célhőmérséklet eléréséhez megfelelően kapcsolja a relét. Természetesen ezt is alaposan megteszteltem, és ahogy látszik a 4.3-mas ábrán, a PI szabályozót úgy hangoltam be, hogy inkább több idő alatt érje el a célhőmérsékletet, de ne legyen túllövés. Azt tekintem úgy, hogy a víz hőmérséklete elérte a cél hőmérsékletet, ha az aktuális hőmérséklet legalább fél fokos környezetében van a cél hőmérsékletnek.

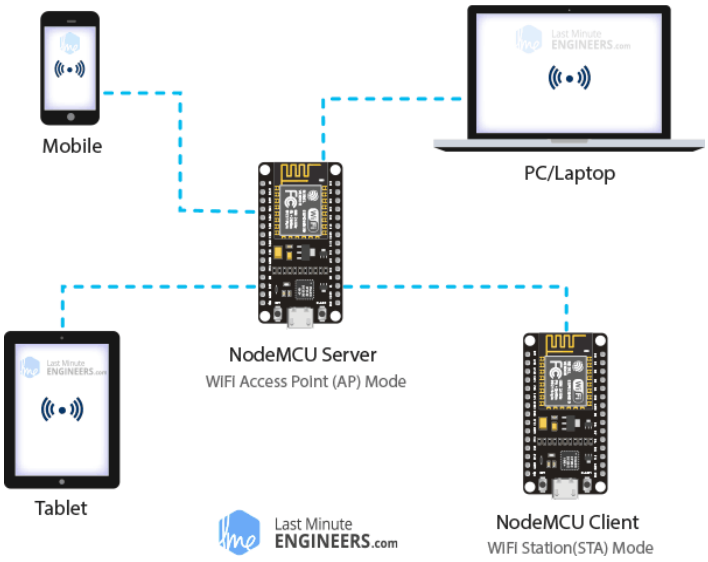


4.3. ábra. A víz hőmérsékletének változása melegítés közben

4.2.4. Webszerver

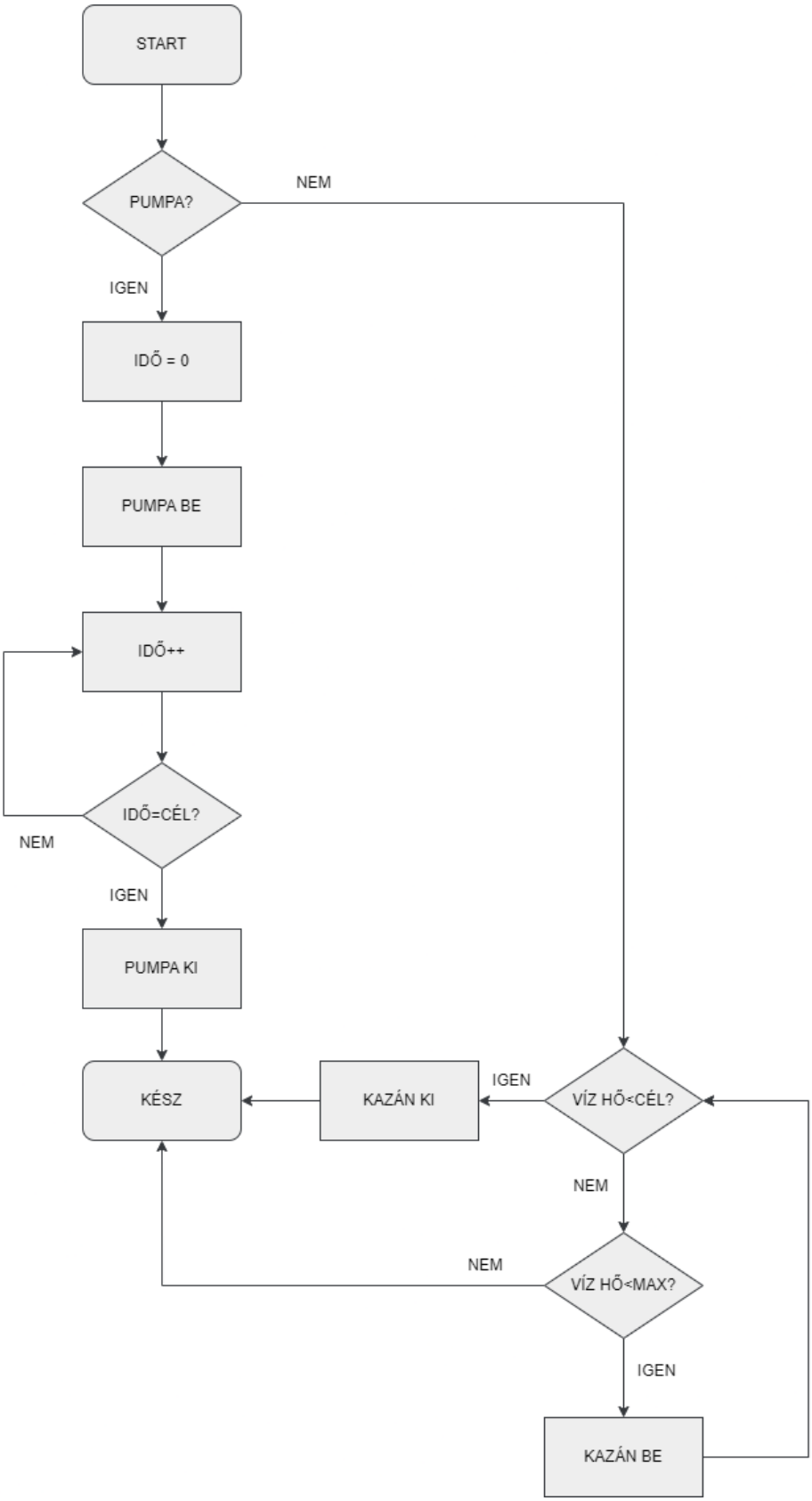
Az ESP8266 chip-el rendelkező fejlesztő modult Soft Access Point (AP) módban használom, amivel saját Wi-Fi hálózatot hozok létre, és ehhez tud maximum 5 kliens csatlakozni. A megadott IP címen lehet webszervert üzemeltetni, amit a hálózatra csatlakoztatott kliensek érnek el. A webszervezen az alábbi végpontokat definiáltam:

- ”/” – gyökér URL
- ”/steps” – a kapott kávéfőzési folyamat lépéseinek belépési pontja. A Json formátumban kapott lista deszerializálása után indul a kávé főzés.
- ”/coffeeDone” – a kávéfőzés állapota kérhető le, FALSE alapértelmezetten, illetve amíg a kávéfőzés tart, TRUE, ha lefőtt a kávé. Szintén Json formátumban küldve.



4.4. ábra. ESP8266 – Soft AP mód (Forrás: [18])

A kliensről kapott lépések listájával dolgozom. Amint sikerül a kapott listát deszerializálni, egy 202-es üzenetet küldök, jelezve a kliensnek, hogy célba ért a konfiguráció, elkezdődött a főzés. A listában meghatározott lépések sorozatából áll össze a kávéfőzési folyamat. Ezeket a lépéseket szekvenciálisan futtatom le egymás után. Miután lefutott az összes lépés, egy *flaget* TRUE-ra állítok és egy 200-as üzenettel elküldöm a szerverre a *flag* értékét Json formátumban. Ezzel jelezve a kliensnek, hogy elkészült a kávé.



4.5. ábra. Egy kávéfőzési konfigurációs lépés blokkdiagramja

4.2.5. Összegzés

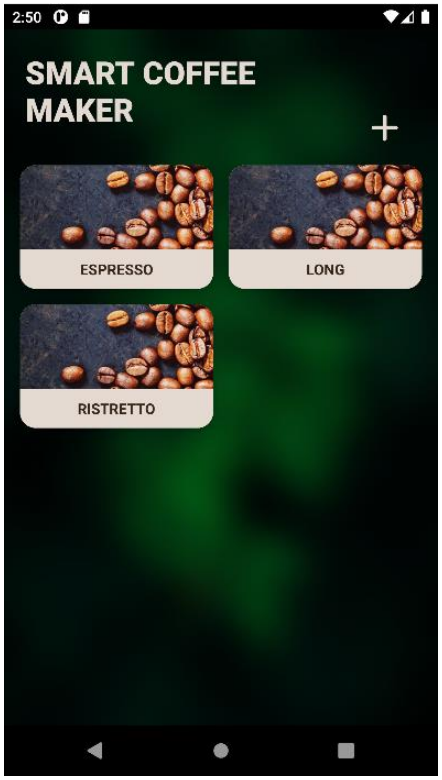
A kávéfőző összeszerelése után képes a rendszer, jelen állapotában csupán böngészőből ugyan, egy kávéfőzési folyamat indítására: Magában foglalva, hogy biztonságos körülmények között a megadott hőmérsékletre képes melegíteni a vizet, és bizonyos átfolyási rátával pumpálni. Mindezt képes egymás után elvégezni minden konfigurált lépésre az alábbi 4.5-ös ábrán szemléltetettek szerint.

4.3. A mobil alkalmazás implementálása

4.3.1. Komponensek

A mobilalkalmazás alapvetően 4 oldalból áll. Minden oldalnak meg van a maga szerepköre. Ezek az oldalak a következők:

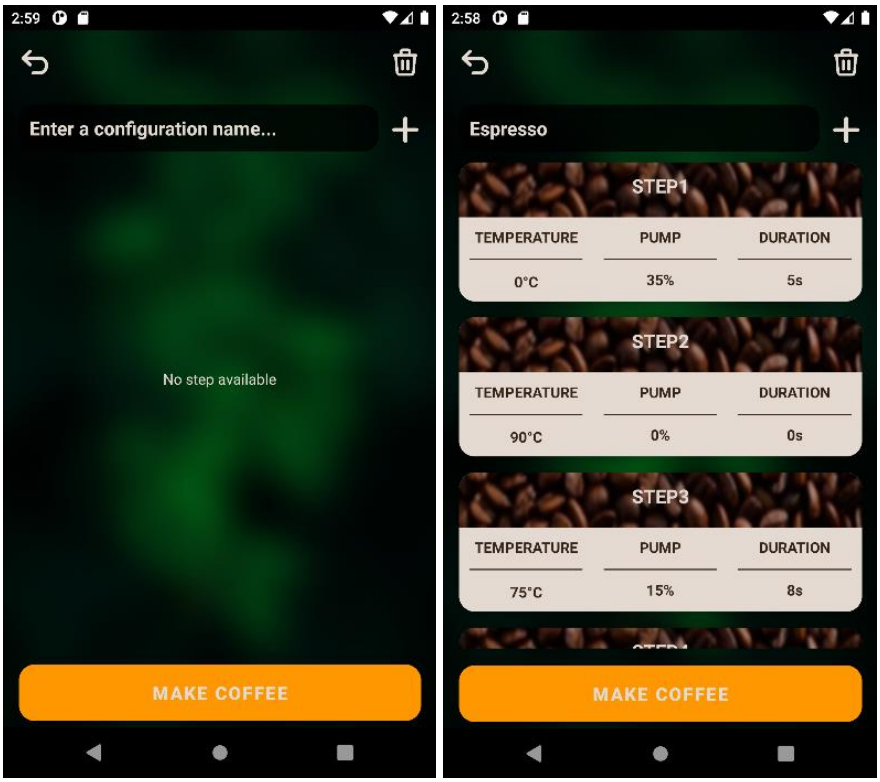
- Főoldal („HomeScreen”)



4.6. ábra. Főoldal („HomeScreen”)

Az alkalmazást megnyitva ez az oldal köszönti a felhasználót. Itt láthatók egy listában a már definiált kávéfőzési konfigurációk. A „+” gombra kattintva megnyílik a konfigurációs oldal („CoffeeConfigScreen”), itt lehet új konfigurációt hozzáadni. Abban az esetben, ha egy lista elemre kattint a felhasználó, akkor a konfigurációs oldalon jelennek meg a kávéfőzési folyamat lépései.

- Konfigurációs oldal („CoffeeConfigScreen”)



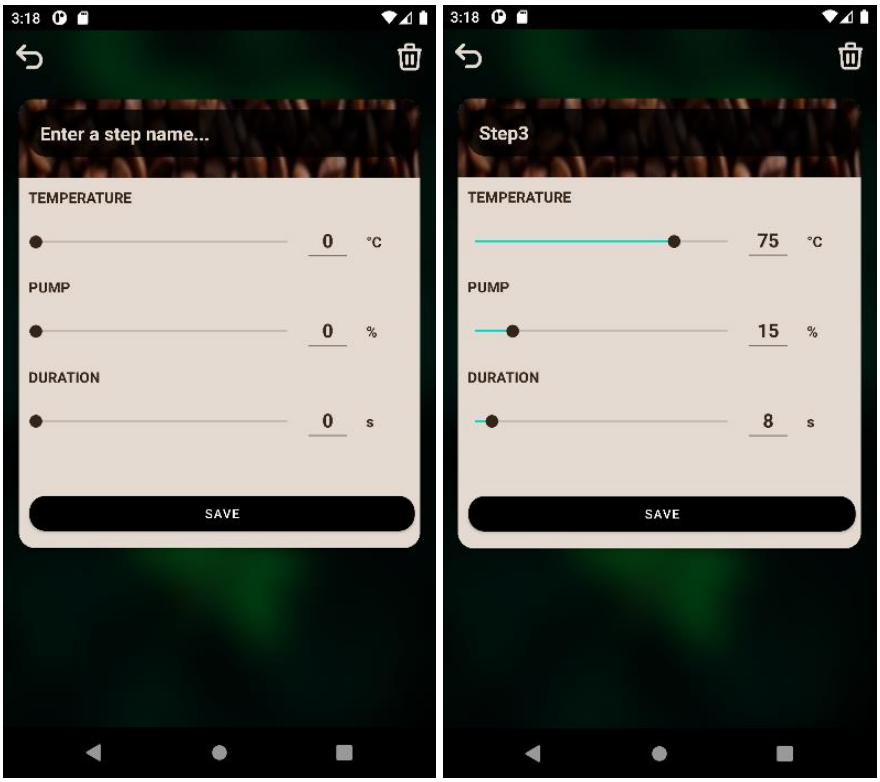
4.7. ábra. Konfigurációs oldal („CoffeeConfigScreen”) új, illetve meglévő elemre

Egy új konfiguráció létrehozásakor egy üres lista jelenik meg. Ekkor a felhasználó megadhat egy nevet, illetve felvehet a listába elemeket, amik a konfiguráció egyes lépései lesznek. Ezt szintén a „+” gomb megnyomásával teheti meg, amivel megnyílik a következő oldal, ahol az egyes lépéseket lehet definiálni.

Abban az esetben, ha meglévő konfigurációt nyitott meg a felhasználó, megjelennek a listában a már definiált lépések. Egy listaelemre kattintva lehet a lépést újra definiálni, emellett lehetőség van a konfiguráció nevét is módosítani.

A jobb felső sarokban lévő „kuka” ikonra kattintva értelemszerűen törölni lehet a megnyitott konfigurációt, a bal felső sarokban lévő „vissza” gombbal pedig vissza lehet navigálni az előző oldalra. A „Make Coffee” gombra kattintva az aktuális kávéfőzési konfigurációt, receptet lehet elindítani. Ekkor a kávéfőzés elindul és megnyílik egy új oldal, ahol a főzés státusza látható.

- Lépés definiáló oldal („EditStepScreen”)

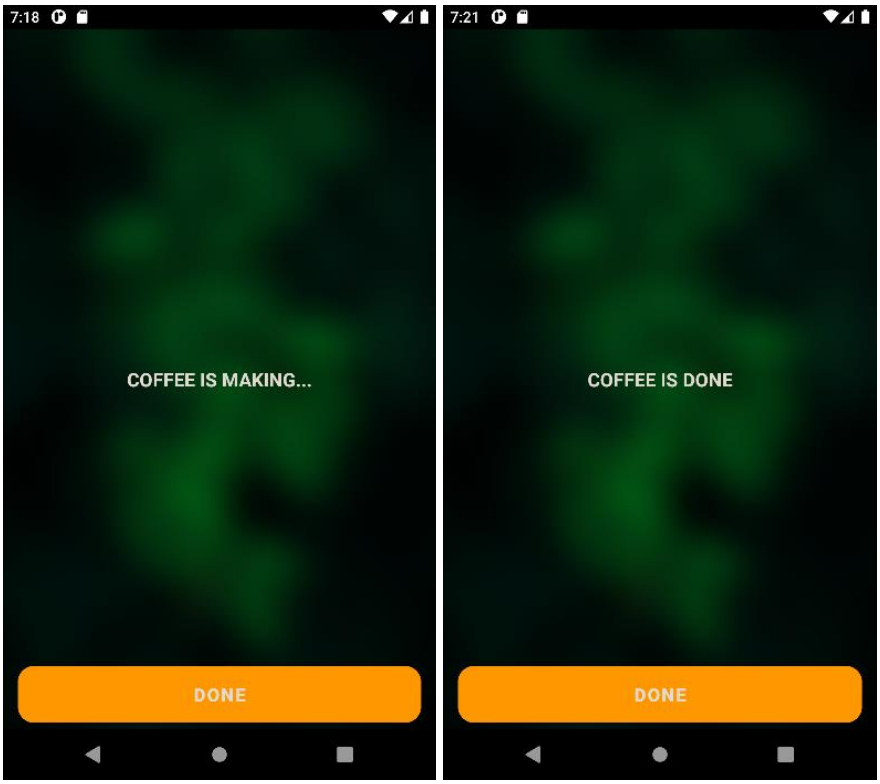


4.8. ábra. Lépés definiáló oldal („EditStepScreen”) új, illetve meglévő elemre

Ha a felhasználó egy új lépést szeretne hozzáadni a konfigurációhoz, akkor itt lehetősége nyílik a lépést elnevezni. A szövegdoboz alatt látható a lépést leíró három tényező. Ezeket a csúszka, illetve billentyűzet segítségével is meg lehet adni. A kettő közül bármelyiket módosítja a felhasználó, a másik is annak alapján állítódik.

A mentés („Save”) gombra nyomva lehet elmenteni a definiált lépést. A nem definiált lépések nem kerülnek elmentésre. A „vissza” gombbal el lehet vetni a módosításokat és így navigál vissza az előző oldalra. A „kuka” ikonra kattintva törölni lehet a lépést.

- Főzés státusz oldal („CoffeeDone”)



4.9. ábra. Főzés státusz oldal („CoffeeDone”) főzés közben, illetve kész állapotban

Ez az oldal nyílik meg, ha elindít a felhasználó egy kávéfőzést. Itt látszik, hogy a kávé éppen készül, és szöveges formában tájékoztat, hogyha elkészült a kávé. A kész (“Done”) gombra nyomva vissza navigál a főoldalra.

Navigáció

4.10. ábra. A mobilalkalmazás navigációs grafikonja

4.3.2. Recept készítés

A felhasználó a “+” gombbal tud új kávéfőzési folyamatot létrehozni. Az új oldalon egy listába lehet felvenni a folyamat lépéseit. Minden lépéshez definiálni lehet a célhőmérsékletet Celsius fokban, százalékos értékben az átfolyási rátát, illetve az időtartamot másodpercben. A felhasználó által konfigurált kávéfőzési folyamatban az alábbi esetek állhatnak elő:

- Célhőmérséklet = 0, átfolyási ráta = 0, időtartam > 0: ekkor az időtartamig vár, a pumpa és a kazán is kikapcsolt állapotban van.
- Célhőmérséklet = 0, átfolyási ráta > 0, időtartam > 0: ekkor az időtartamig, az átfolyási rátával bekapcsol a pumpa, kazán kikapcsolt állapotban van.
- Célhőmérséklet > 0, átfolyási ráta = 0, időtartam = 0: ekkor a kazánban lévő vizet a célhőmérsékletig melegíti, a pumpa kikapcsolt állapotban van.
- Célhőmérséklet > 0, átfolyási ráta = 0, időtartam > 0: ekkor a célhőmérsékletet tartja az időtartamig, a pumpa kikapcsolt állapotban van.
- Célhőmérséklet > 0, átfolyási ráta > 0, időtartam > 0: ekkor a célhőmérsékletet tartva, az időtartamig, az átfolyási rátával működik a pumpa.

4.3.3. Kommunikáció

A kliens oldali kommunikáció a Retrofit2 segítségével történik. Ez egy REST kliens, amelynek célja, hogy könnyebbé tegye a REST szolgáltatások használatát. A kliens egy API-n keresztül kommunikál a szerverrel. Két féle http hívásra van szükség a kommunikáció során. Az egyik egy POST hívás, amely segítségével küldöm az összeállított kávéfőzési konfigurációt a szervernek, a másik pedig egy GET hívás, amivel a kávéfőzés státuszát kérem le a szerverről.

Kávéfőzési konfiguráció küldése a szerverre

Ahogy korábban már említettem, egy kávéfőzési konfiguráció lépések listájából áll. Kód szinten is így van eltárolva objektumonként. Minden ilyen lépés objektumnak van négy adattagja: a lépés elnevezése, a célhőmérséklet, az átfolyási ráta, és az időtartam. Ezeket a lépéseket kell egy POST hívásban elküldeni a szervernek. Ehhez szerializálni kell az objektumokat, amihez tökéletes választás a GSON. Nagyon egyszerű a használata, egy metódushívással képes az egész objektum listát Json formátumúvá alakítani. Ezt a Jsont küldöm fel a szerverre a "/steps" végpontra, ahol a weboldal törzsébe kerül. Ezt a mikrokontroller oldalán elérem és ott történik az adatfeldolgozás.

A kávéfőzés állapotának lekérése

Miután elküldtük a szervernek a kávéfőzési konfigurációt, automatikusan elindul a főzés. Ezután GET kérést küldök a szerverre, hogy lekérjem a kávéfőzés állapotát. Amint egy 200-as üzenetet kapok, lekérem a kávéfőzés állapotát a weboldal törzséből. Ezt szintén Json formátumban kapja meg a kliens, majd deszerializálom és megjelenítem a felhasználói felületen.

Mentés és CRUD műveletek

Most, hogy elkészült az alkalmazás, és kiépült a kommunikáció a webszerverrel, a következő lépés a mentés implementálása. A kávéfőzési konfigurációk listáját a telefon tárhelyén fogom tárolni, amihez egy úgynevezett *SharedPreferences*-t fogok használni. Ez egy interfész, amely lehetővé teszi, hogy tároljunk, módosítsunk, töröljünk lokálisan adatot. Ehhez készítettem egy fájlt, amiben tárolni fogom a *SharedPreferences* adatot. Ennek az egyszerűsége abban rejlik, hogy primitív típusokat képes tárolni, ezért ismét a jól bevált GSON-t használtam arra, hogy az egész listát, ami kávéfőzési konfigurációkból áll, Json formátumúvá alakítsam. Az így kapott szöveges forma már megfelelő ahhoz, hogy *SharedPreferences*-ben tároljam.

Létre hoztam és elmentettem néhány alapvető, általam összerakott kávéfőzési konfigurációt, ezek az alkalmazás indításakor betöltődnek a felhasználói felületre. A problémát itt az jelentette, hogy nem volt egyértelmű, hogy hol érdemes menteni a CRUD

műveletek hatására. Ezt úgy oldottam meg, hogy amikor egy új konfigurációt létrehoz a felhasználó, a „+” gomb nyomása után elmentek egy új konfigurációt, de egy üres lépés listával. A lépések definiálása után a mentés gomb nyomását követően visszanavigál a konfigurációs oldalra, és megtörténik a lépés mentése is. Ha a főoldalra is visszanavigál a felhasználó, akkor mindig az aktuális legfrissebb elmentett adat töltődik be. A törlés rá van kötve a felhasználói felületen található „kuka” gombokra, ezzel a *SharedPreferences*-ből is törlődnek a kívánt adatok. Módosítás esetén pedig felülírom az előző verziót.

4.3.4. Összegzés

Tehát kliens oldalon is sikerült kiépíteni egy kommunikációt a webszerverrel, amelyen keresztül lehet egy kávéfőzési konfigurációt küldeni, illetve állapotot fogadni. Ezen felül a mobil alkalmazás is implementálásra került, a felhasználó képes CRUD műveleteket végrehajtani, mentésre kerülnek az adatok a felhasználó telefonján, és Wi-Fi hálózatra csatlakozva ezeket elküldve kávéfőzést is tud indítani vezeték nélküli kapcsolattal. Végül a főzés befejeztével értesül a felhasználó arról, hogy kész a kávé.

5. TESZTELÉS

5.1. A hardveres komponensek működésének validálása

5.1.1. A szivattyú tesztelése

A pumpa teljesítménye nyilván teljesen más egy rendszerben, mint önmagában. Így a már kész kávéfőző pumpáját teszteltem, hogy adott idő alatt, különböző kitöltési tényezővel mekkora mennyiségű vizet pumpál ki a kifolyócsőrön keresztül. Először teszteltem véletlenszerű sorrendben, üres portafilterrel, kávé nélkül. A kazán végig kikapcsolt állapotban volt.

Átfolyási ráta	30 sec
10%	17 g
20%	35 g
30%	50 g
40%	65 g
50%	77 g
60%	95 g
70%	100 g
80%	120 g
90%	135 g
100%	140 g

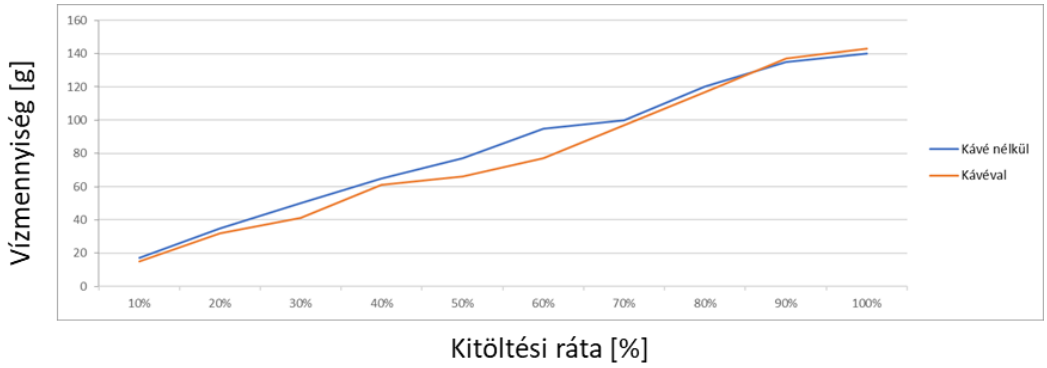
5.1. táblázat. Pumpa tesztelése üres portafilterrel

Második körben pedig ugyanezekkel a paraméterekkel teszteltem a pumpa teljesítményét 10g kávéval és megvizsgáltam, hogy így változik-e, és ha igen mennyivel, a rendszeren átpumpált víz mennyisége.

Átfolyási ráta	30 sec
10%	15 g
20%	32 g
30%	41 g
40%	61 g
50%	66 g
60%	77 g
70%	97 g
80%	117 g
90%	137 g
100%	143 g

5.2. táblázat. Pumpa tesztelése kávéval

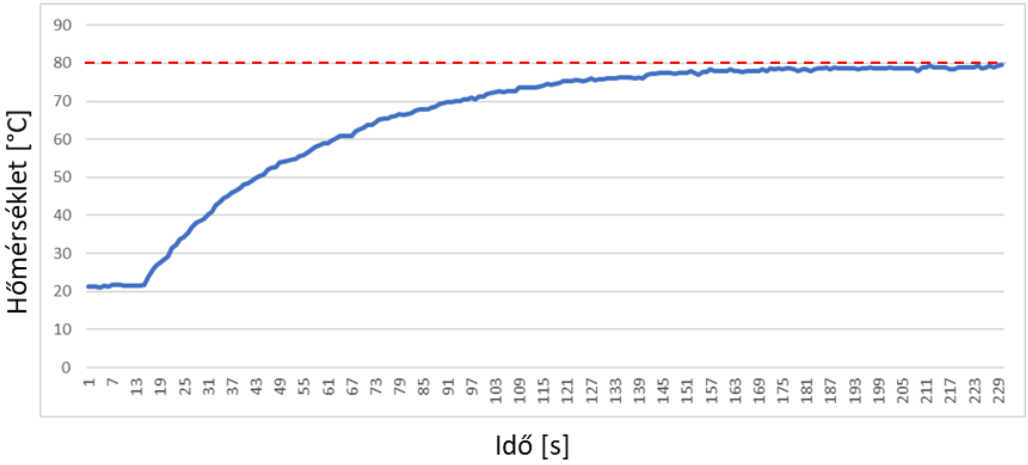
Összehasonlítva a két eredményt az látszik, hogy nagyon kis eltérés van alacsony átfolyási rátával, de valamivel kevesebb víz folyik át, ha kávé is van benne. Azonban 20% és 70%-os kitöltési ráta között viszonylag nagy eltérés tapasztalható. Egyértelműen kevesebb vizet pumpál át a rendszeren kávéval, mint anélkül. Viszont 70%-os átfolyási rátával, vagy nagyobbal újra alig észlelhető a különbség, szinte elhanyagolható.



5.1. ábra. Pumpa tesztesetek összehasonlítása

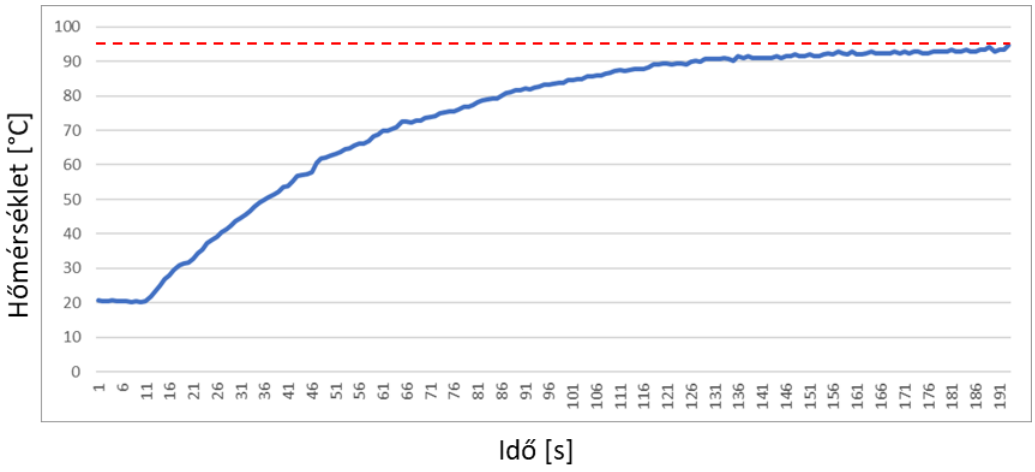
5.1.2. A kazán tesztelése

A kazán esetében azt teszteltem, hogy milyen gyorsan képes felfűteni a vizet, ha nem jár a pumpa. Két különböző célhőmérséklettel vizsgáltam. Az egyik 80 °C volt és feljegyeztem közben a mért hőmérséklet értékeket.



5.2. ábra. Kazán tesztelése 80 °C célhőmérséklettel

A tesztelés során 21 °C-ról indult a víz hőmérséklete, az elején megfigyelhető egy kis hiszterézis, hiszen kb. csak 15 másodperc múlva történik tényleges hőmérsékletnövekedés, majd hirtelen nagyon megindul fölfelé a víz hőmérséklete és ugyan hosszú idő alatt, de szépen beáll 80 °C-ra. Ez annak köszönhető, hogy biztonsági szempontokat figyelembe véve a PI szabályozót úgy hangoltam be, hogy ne legyen túllövés, még ha az a sebesség rovására is megy. Így összesen 230 másodpercet vett igénybe a melegítés 21 °C-ról 80°C-ra.



5.3. ábra. Kazán tesztelése 95 °C célhőmérséklettel

A második teszt során 20 °C-ról indult a víz hőmérséklete és 95 °C volt a célhőmérséklet. Itt is hasonlóan az előzőhöz, megjelenik hiszterézis az elején, de valamivel gyorsabban elindul a hőmérséklet növekedés. Meglepődve tapasztaltam, hogy jelen esetben hiába nagyobb volt a kezdetleges hőmérséklet különbség, mégis majdnem 40 másodperccel hamarabb elérte a cél hőmérsékletet, mint az előző tesztesetben. Így összesen 193 másodpercig tartott felfűtenie a kazánnak a vizet 20 °C-ról 95 °C-ra.

5.2. Az applikáció működésének validálása

5.2.1. Recept készítés

A következő egy manuális teszt volt. Megvizsgáltam, hogy a mobil alkalmazás egy recept elkészítésénél az elvártnak megfelelően működik-e. A teszt részletes leírása az 5.3-mas táblázatban található.

Teszteset	Teszt eredmény
1. Alkalmazás megnyitása	✅
2. „+” gomb megnyomása	✅
Elvárás: Megnyílik egy új oldal, a lista és a név üres	✅
3. Konfiguráció elnevezése	✅
4. „+” gomb megnyomása	✅
Elvárás: Megnyílik egy új oldal, a lépés neve üres, a hőmérséklet, pumpa és időtartam értékei alapértelmezetten 0.	✅
5. Lépés elnevezése	✅
6. Hőmérséklet, pumpa és időtartam beállítása	✅
7. „Save” gomb megnyomása	✅
Elvárás: Navigáció vissza a konfigurációs oldalra. A listában szerepel a létrehozott lépés, a megadott beállításokkal és a nevével.	✅
8. Tetszőleges számú lépés létrehozása a 4-7. pont ismétlésével.	✅
Elvárás: A listában szerepel az összes létrehozott lépés, a megadott beállításokkal és a nevével a hozzáadás sorrendjében.	✅
9. „vissza” gomb megnyomása	✅
Elvárás: Navigáció a főoldalra. A listában szerepel a létrehozott konfiguráció, a neve megfelelő.	✅

5.3. táblázat. Recept készítés tesztelése

5.2.2. Az applikáció és a hardver kommunikációja

Mi történik, ha főzés közben megszakad a Wi-Fi kapcsolat?

Abban az esetben, ha elindította a felhasználó a telefonjáról a főzési konfigurációt, az azt jelenti, hogy az összes lépés egy POST kéréssel elment a szervernek. Innentől kezdve, ha bármikor megszakad a Wi-Fi kapcsolat, az nincsen befolyással a kávéfőzés menetére.

Legrosszabb esetben, ha nem kapcsolódik újra a kávéfőzés befejeztéig, akkor nem kap értesítést a felhasználó arról, hogy kész van a kávé.

Mi történik, ha főzés közben kikapcsolom a kávéfőzőt?

Mivel a mikrokontrollernek szüksége van 5V-ra, a relére pedig rá van kötve a 230V egyik vezetéke a kapcsoláshoz, ezért, ha ezt a tápfeszültséget lekapcsolom, akkor a mikrokontroller továbbra is feszültség alatt marad. Így tehát az tovább működik, kapcsolja a kapott konfigurációnak megfelelően a relét, azonban mivel nincs feszültség alatt, így a kávéfőző ebben az állapotban nem működik. Ha ez főzés közben történik, akkor félbe marad a kávéfőzés.

5.3. Gyakorlati alkalmazhatóság vizsgálata

Az eddigi tesztek során azt tapasztaltam, hogy a 95 °C körüli hőmérséklet elérése közben nem elhanyagolható mennyiségű víz folyik át a portafilteren. Ezért hasznosnak érzem egy olyan teszt elvégzését, mikor 95 °C-ra melegítem a vizet és megmérem a közben kifolyt víz mennyiségét. A cél az, hogy kalkulálni lehessen, hogy mekkora mennyiségű legyen a kész kávé. A teszt során 10g kávét használtam, és a célhőmérséklet 95 °C, a pumpa kikapcsolt állapotban van.

Teszt	Víz
1.	11 g
2.	7 g
3.	8 g
4.	7 g
5.	6 g

5.4. táblázat. Kifolyt víz mennyisége 95 °C-ra melegítés közben

A mért értékek első ránézésre nagyon kevésnek tűnnek, viszont egy eszpresszó kávé mennyiségre kb. 30-50 ml. Ehhez képest az átlagos kb. 8 g víz nem elhanyagolható, tehát ahhoz, hogy pontosan konfigurálni lehessen a kávéfőzési folyamatot, akkor ezzel számolni kell.

Kávé készítés

Most, hogy minden szükséges eszköz és tudás a birtokomban áll, készítsünk kávét. A következő teszt során készítek egy nem ideális beállításokkal készült kávéfőzési konfigurációt, illetve egyet optimális beállításokkal és ezeket összehasonlítom. Ebben az esetben is szintén 10 g eszpresszó pörkölésű és őrlésű kávét használtam.

A nem ideális konfiguráció a következő:

1. lépés

   Átfolyási ráta = 0 %
   Cél hőmérséklet = 95 °C
   Időtartam = 0 s

2. lépés

   Átfolyási ráta = 20 %
   Cél hőmérséklet = 95 °C
   Időtartam = 10 s

3. lépés

   Átfolyási ráta = 0 %
   Cél hőmérséklet = 92 °C
   Időtartam = 30 s

4. lépés

   Átfolyási ráta = 10 %
   Cél hőmérséklet = 92 °C
   Időtartam = 5 s

5. lépés

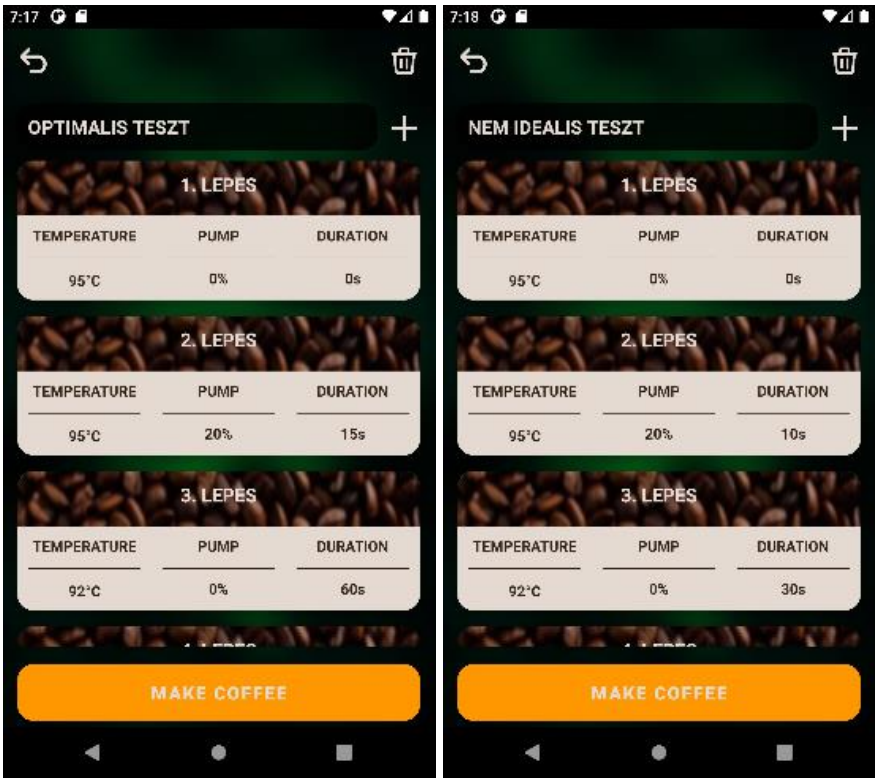
   Átfolyási ráta = 100 %
   Cél hőmérséklet = 92 °C
   Időtartam = 6 s

Az elvárt mennyiség: 8 g (1. lépés) + ~10,6 g (2. lépés) + 2,5 g (4. lépés) + 28,6 g (5. lépés) = 49,7 g

Tényleges mennyiség: 46 g

Ez a konfiguráció elsőre ideálisnak tűnhet, viszont figyelembe kell venni azt is, hogy a kazánt követő csőben is lehet víz, így a 2. lépésben definiált átfolyási ráta az időtartammal kevésnek bizonyult ahhoz, hogy a forró víz elérje a kávét. Ezzel együtt kevésnek bizonyult az előáztatás ideje, amit a 3. lépésben definiáltam. Végül a 4. lépésben túl alacsony, az 5. lépésben pedig túl magas átfolyási rátát használtam, ezért a végeredmény sokkal inkább egy teára hasonlított, mintsem kávéra. A színe áttetsző volt, inkább sárgásbarna, víz állagú, a kávé ízét csak nyomokban lehetett érezni. Mennyiségét tekintve

valamivel kevesebb, mint az elvárt volt, viszont, ha figyelembe veszem, hogy nagyon kis átfolyási rátákat használtam, kis időtartammal, így nem annyira meglepő és nagyságrendileg elfogadhatónak vélem.



5.4. ábra. Egy optimális és egy nem ideális konfiguráció a mobil alkalmazásban

A következő teszteset pedig egy ideális főzési konfigurációval készült. Itt már figyeltem a kávéfőző tulajdonságaira, viszont ugyanazt a kávét használtam, mint az előbb, szintén 10 g mennyiséggel.

Az optimális beállítással készült konfiguráció a következő:

1. lépés

   Átfolyási ráta = 0 %
   Cél hőmérséklet = 95 °C
   Időtartam = 0 s

2. lépés

   Átfolyási ráta = 20 %
   Cél hőmérséklet = 95 °C
   Időtartam = 15 s

3. lépés

   Átfolyási ráta = 0 %
   Cél hőmérséklet = 92 °C
   Időtartam = 60 s

4. lépés

   Átfolyási ráta = 30 %
   Cél hőmérséklet = 92 °C
   Időtartam = 15 s

Az elvárt mennyiség: 8 g (1. lépés) + ~16 g (2. lépés) + 20,5 g (4. lépés) = 44,5 g

Tényleges mennyiség: 47 g

A hosszabb előáztatás hatására az így készült kávé színe sokkal sötétebb, markáns sötétbarna színű, az állaga is jóval testesebb, nem annyira vizes, valamint ízre is sokkal élvezhetőbb és erősebb is. A lefőtt kávé mennyiségét tekintve ebben az esetben enyhén több lett, mint az elvárt, de ez is azon a határon belül van, amit még abszolút elfogadhatónak tekintek.

6. EREDMÉNYEK, KÖVETKEZTETÉSEK, TOVÁBBFEJLESZTÉSI LEHETŐSÉGEK

A dolgozatom során sikerült egy régi, olcsó kávéfőzőt szétszerelni, tanulmányozni a felépítését és működését. Az így szerzett kávéfőző elemekből és az általam választott komponensek segítségével megterveztem és összeraktam egy működő kávéfőzőt. Majd a szakdolgozatom keretein belül fejlesztettem egy mobil alkalmazást hozzá, mellyel a felhasználó vezérelni tudja a kávéfőzőt. Kávéfőzési konfigurációkat képes kezelni, amiket a felhasználó vezeték nélküli kapcsolaton keresztül tud a kávéfőzőre küldeni és végrehajtani. A konfiguráció teljes körű személyre szabhatósága miatt, bizonyos keretek között, de csak a felhasználó kreativitása és tudása szab határt.

A tesztelések során abszolút beigazolódott, hogy ugyanazzal a kávéfőzővel és kávéval teljesen más kávé főzhető. Hatalmas különbség volt az optimális és a nem ideális konfigurációval főzött kávék között. Ebből arra tudok következtetni, hogy „jó" és „jobb" kávéfőzési konfiguráció is létezik, csupán a kávéhoz és a kávé főzéshez kell érteni. Egy hozzáértő baristának számtalan lehetőséget nyújthat a kísérletezéshez, vagy akár a kávék tökéletesítéshez.

Továbbfejlesztési lehetőségek

- Víz mennyiségének meghatározása

  Az átfolyási ráta százalékos meghatározása nehézkessé teszi a lefőtt kávé mennyiségének meghatározását. Ezen javítani lehet azzal, ha beszerelek egy áramlás mérő szenzort a pumpa és a kazán közé. Így meg lehetne valósítani azt, hogy milliliter pontossággal lehessen állítani, hogy a pumpa mennyi vizet adagoljon időegység alatt. Alapvetően két átfolyásmérő szenzor található a piacon, amelyek kifejezetten kávégépekhez készültek. A legnagyobb különbség köztük, hogy az egyiknek műanyag, míg a másiknak rozsdamentes acél háza van. Ebből következik, hogy természetesen a műanyag nem bírja annyira a hőt, kb. 100 °C a tűréshatára, ami nagyon a határán van a kávé ideális főzési hőmérsékletének. Ez a probléma kiküszöbölhető, amennyiben a szenzor a kazán előtt helyezkedik el. Ezzel szemben a rozsdamentes acél változata könnyedén megbirkózna a szükséges főzési hőmérséklettel, azonban ennek az ára a műanyag verziónak kb. az ötszöröse.

- Kávéfőzés ütemezése

  Ebben a rohanó, automatizált világban az ember mikor reggel fölkel, csak arra vágyik, hogy a frissen főtt kávé várja, és így biztosan jól induljon a napja. Erre megoldást nyújthat egy olyan funkció, melynek segítségével ütemezni lehet a kávé főzés indítását. Ehhez plusz egy idő paramétert rendelnék a kávéfőzési

konfigurációhoz, ezt a szerverre küldve a meghatározott időben indulna el a kávéfőző a kapott konfigurációval.

- NodeMCU szerver Wi-Fi (STA) módban

Az STA mód azt jelenti, hogy a mikrokontroller kliensként szerepelne a hálózaton, így csatlakozni tud az otthoni Wi-Fi hálózatra. A jelenlegi működéssel ugyanis telefonnal közvetlenül a mikrokontrollerhez lehet csatlakozni Wi-Fi-n keresztül, tehát ekkor nem tud a felhasználó a telefonján internetet használni Wi-Fi-n keresztül. Erre lenne megoldás az STA mód, mellyel megvalósítható, hogy a mikrokontroller és a mobil telefon kommunikációja az otthoni hálózaton, *routeren* keresztül valósuljon meg. Így a felhasználónak nem kell a telefonján hálózatot váltani, ha használni szeretné a mobil applikációt.

- Komolyabb kávéfőző

Korábban már említésre került, hogy a dolgozatom egyik célja, hogy egy olyan kávéfőzőt és egy hozzá tartozó alkalmazást implementáljak, amely minimális módosítással átültethető egy komolyabb kávéfőző gépbe. Biztos vagyok benne, hogy egy újabb, jobb eszpresszó kávéfőző gépbe átültetve a megvalósításomat nagyobb eredményeket lehetne elérni a kávé minőségét illetően.

6.1. ábra. Kész kávéfőző és mobilalkalmazás

7. ÖSSZEGZÉS

A szakdolgozatom célja egy mobilalkalmazás fejlesztése volt, amelyen keresztül a felhasználó képes egy kávéfőzőt vezérelni. Az alkalmazás felhasználói célpontjai olyan hozzáértő baristák, akik kellő szakértelemmel rendelkeznek, hogy az alkalmazásban létrehozott kávéfőzési konfigurációval a kávé minőségét javítsák. Utána néztem, hogyan működnek általánosságban az eszpresszó kávéfőzők, ezután tanulmányoztam a kávéfőzőt, amivel dolgoztam. Megvizsgáltam hogyan működnek a rendelkezésre álló komponensek és összeszedtem milyen alkatrészekre lesz szükség.

Specifikáltam a mikrokontroller oldaláról a szükségleteket és meghatároztam, hogy a szoftver milyen funkciókat kell ellásson. A részletes specifikáció során megfogalmaztam, hogy milyen kritériumoknak kell megfeleljen a kész rendszer, döntöttem a vezeték nélküli technológiáról, a mikrokontrollerről, valamint kiválasztottam a mobil platformot, a fejlesztő környezetet, illetve a programozási nyelvet.

A tervezést a biztonsági oldallal kezdtem felhasználói, illetve fejlesztői oldalról. Majd megterveztem a hardveres részét, ennek két fő komponense a pumpa és a kazán volt. Már ezen a ponton is sok tesztelésre volt szükség, hiszen nem volt elérhető számomra a komponensek gyártói adatlapjai. Megvalósítottam egy saját PWM-et, rákötöttem a relére a pumpát és a kazánt, majd implementáltam a PI szabályozót és behangoltam. Kialakítottam a kommunikációt a szerver és a kliens között, összeraktam a kávéfőzőt és következett a mobil alkalmazás tervezése és fejlesztése. A kliens oldali kommunikáció implementálásával kezdtem, majd megcsináltam az alkalmazás felhasználói felületét, létrehoztam az oldalakat és a navigációt közöttük. Végül a CRUD műveleteket és a mentést is implementáltam.

Az elkészül kávéfőzőt több szempontból is teszteltem. A hardveres komponensekkel kezdtem. A pumpa esetén megvizsgáltam, hogy adott idő alatt különböző kitöltési rátával hogyan viselkedik kávéval, illetve anélkül. A kazán esetében pedig megmértem, hogy mennyi idő alatt képes felmelegíteni a vizet különböző hőmérsékletekre. A mobil alkalmazást manuálisan teszteltem, illetve megvizsgáltam a kommunikációt szerver és kliens között, hogy hogyan viselkedik különböző nem várt helyzetekben.

A munkám eredményeként elkészült egy mobil alkalmazás, amiről vezérelni lehet az általam épített kávéfőzőt. A fejlesztés során törekedtem arra, hogy a kész alkalmazás egyszerű legyen, letisztult, felhasználó barát és használható. Így az androidos készülékemről már tudok kávéfőzési konfigurációt összeállítani, kísérletezni új kávékkal, ízekkel. A tesztelés során pedig bebizonyosodott, hogy valóban növelhető a kávé minősége, ha csupán a kávéfőzési konfigurációt változtatjuk, így tehát a szakdolgozatom elérte célját.

8. SUMMARY

The goal of my thesis was to develop a mobile application, through which the user is able to control a coffee machine. The target users of the application are competent barists who have the expertise to improve the coffee quality using coffee brewing configurations created in the application. I researched how coffee machines usually work, then I studied the coffee machine that I was working with. I examined how the available components work.

I specified the needs from the microcontroller's side to decide which functions should the software have. During the detailed specification I drafted the criterias which the system must meet with and I decided about the wireless technology and the microcontroller. I chose the mobile platform, the development framework and the programming language.

I started planning the security side from the user's and the developer's perspective. Then I planned the hardware which has two main components, the pump and the boiler. A lot of tests were needed to be carried out at this point because the manufacturer's data sheets of the components were not available. I implemented my own PWM, then I connected the pump and the boiler to the relay. I created a PI controller and I tuned it. I established the communication between the server and the client then built the coffee machine. The next step was to plan and develop the mobile application. I started with the communication on the client's side then I created the user interface of the application including the pages and the navigation. Finally, I implemented the CRUD operations and the saving.

I tested the completed coffee maker from several perspectives starting with the hardware components. In case of the pump I analized how it behaves with and without coffee with a given duration and with different duty cycles. In case of the boiler I measured how much time is needed to heat the water to the given temperature. I tested the mobile application manually, and I also tested how reliably the communication works in unusual situations.

As a result of my work a mobile application has been created which can control the coffee machine. During development I was focusing on the application to be simple, clean, user friendly and usable. Since then I can create coffee brewing configurations from my android device, I can experiment with new coffee types and flavors. During the testing it was proven that the quality of the coffee can be increased by changing the coffee brewing configuration, so my thesis achieved its goal.

9. IRODALOMJEGYZÉK

[1] Paul Paquin: Functional and speciality beverage technology. Woodhead Publishing Limited and CRC Press LLC, 2009

[2] Szuna Noémi: Csészével a világ körül. Central Kiadó, 2019

[3] Starbucks Coffee Company: Coffee Master. Starbucks Coffee Company, 2006

[4] Baggenstoss J et al.: Coffee roasting and aroma formation: application of different time-temperature conditions. J Agric Food Chem. 56(14), 2008

[5] Michael I. Cameron et al.: Systematically Improving Espresso: Insights from Mathematical Modeling and Experiment. Matter, Vol 2, Issue 3, 2020.

[6] Clive Coffee: How Do Eszpresszó Machines Work?, (https://clivecoffee.com/blogs/learn/how-do-eszpresszó-machines-work), utoljára megtekintve: 2022. 04. 08.

[7] Salim Jibrin Danbatta, Asaf Varol: Comparison of Zigbee, Z-Wave, Wi-Fi, and Bluetooth Wireless Technologies Used in Home Automation. IEEE, 7th International Symposium on Digital Forensics and Security (ISDFS), 2019

[8] Okosotthon bolt.hu: Mi az a Zigbee és miért jó az okosotthonosnak? (https://okosotthon.bolt.hu/webaruhaz/okosotthon-kisokos/mi-az-a-zigbee-es-miert-jo-az-okosotthonodnak/), utoljára megtekintve: 2022. 04. 18.

[9] Espressif: ESP8266 Technical Reference (https://www.espressif.com/sites/default/files/documentation/esp8266-technical_reference_en.pdf), utoljára megtekintve: 2022. 04. 08.

[10] TESLA Living Systems: Water Pump With PWM Speed Control And Flow Sensor (https://www.tesla.eu.com/2020/12/18/controlling-water-pump-and-measuring-flow-sensors/), utoljára megtekintve: 2022. 04. 19.

[11] LF Group: Spare parts for: Eszpresszó machines (https://www.atp-czesci.pl/czesci_wg_producentow/producenci/ekspresy_producenci/ULKA.pdf), utoljára megtekintve: 2022. 05. 07.

[12] ElectroNoobs: Temperature PID controller – Arduino (http://electronoobs.com/eng_arduino_tut24_2.php), utoljára megtekintve: 2022. 04. 19.

[13] Thomas J. Cortina, Ellen L. Walker, Laurie Smith King, David R. Musicant: SIGCSE '11: Proceedings of the 42nd ACM technical symposium on Computer science education. Association for Computing Machinery, 2011

[14] SmartKit: Mi a különbség a Z-Wave és a Zigbee között? (https://www.smartkit.hu/mi-a-kulonbseg-a-z-wave-es-a-zigbee-kozott-132), utoljára megtekintve: 2022. 04. 18.

[15] Federica Laricchia: Mobile operating systems' market share worldwide from January 2012 to January 2022

[16] Subham Bose, Madhuleena Mukherjee, Aditi Kundu, Madhurima Banerjee: A comparative study: java vs kotlin programming in android application development. International Journal of Advanced Research in Computer Science, 3 (9), 2018

[17] Conrad Electronic SE, Klaus-Conrad-Str. 1, D-92240 Hirschau: BN 2268118 5 V 2-Channel Relay Module for Arduino (https://asset.conrad.com/media10/add/160267/c1/-/en/002268118ML00/hasznalati-utmutato-2268118-tru-components-tc-9072472-rele-panel-1-db-alkalmas-arduino.pdf), utoljára megtekintve: 2022. 09. 15.

[18] LastMinuteEngineers.com, Create A Simple ESP8266 NodeMCU Web Server In Arduino IDE (https://lastminuteengineers.com/creating-esp8266-web-server-arduino-ide/), utoljára megtekintve: 2022. 12. 03.

10. ÁBRAJEGYZÉK

11. TÁBLÁZAT JEGYZÉK

12. RÖVIDÍTÉSEK

BLE	Bluetooth Low Energy
CLI	Command-Line Interface
CRUD	Create-Read-Update-Delete
DC	Direct Current
EEPROM	Electrical Erasable Programmable read-only Memory
GPIO	General Purpose Input/Output
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers
MCU	Micro Controller Unit
OS	Operating System
PID	Proportional-Integral-Derivative
PWM	Pulse Width Modulation
REST	Representational State Transfer
SDK	Software Development Kit
SRAM	Static Random Access Memory
URL	Uniform Resource Locator
USB	Universal Serial Bus
WEP	Wired Equivalent Privacy
WPA	Wi-Fi Protected Access