

Fejlesztői dokumentáció

Vidics Márk

T1YAAB

Mérnökinformatikus BSc

Tartalomjegyzék

1. A projekt áttekintése	2
1.1. Funkcionális követelmények	2
1.2. Nem funkcionális követelmények	2
2. Tervezési fázis	3
2.1. Hardveres tervek	3
2.2. Szoftveres tervek	3
3. Hardver specifikáció	4
3.1. Próbapanel	4
3.2. Raspberry Pi	5
3.3. RFID olvasó	6
3.4. LCD kijelző	8
4. Szoftver specifikáció	9
4.1. Raspberry Pi OS	9
4.1.1. Docker	10
5. A fejlesztett kód részletezése	10
5.1. Adatbázis szerkezet	10
5.2. Verziókövető rendszer	12
5.3. Konfiguráció	12
5.4. Indító szkript	13
5.5. SPI kommunikáció	13
5.6. I ² C kommunikáció	13
5.7. Email küldés	13
5.8. HTTP hívások	13

Táblázatok jegyzéke

1. Az RFID olvasó bekötési terve	8
2. Az LCD kijelző felhasznált lábai	9
3. Az LCD kijelző bekötési terve	9

4.	Az felhasználókkal kapcsolatos alapinformációk tárolása	11
5.	Az RFID kulccsal kapcsolatos alapinformációk tárolása	11
6.	Egy felhasználó összekapcsolása több RFID kulccsal (egy-a-többhöz)	11
7.	Egy felhasználóhoz tartozó RFID kulcs belépési idejének tárolása	12

1. A projekt áttekintése

1.1. Funkcionális követelmények

- A projekt célja egy olyan bemutató berendezés elkészítése, amely segítségével képesek vagyunk belépési eseményeket megjeleníteni, illetve karbantartani RFID kulcsok segítségével.
- Szükségünk van egy kártyaolvasó eszközre, amely a felhasználóknak kiosztott kártyák/kulcsok értékét beolvasni.
- Képesnek kell lennünk ezeket a kártyákat felhasználókhoz társítanunk.
- A rendszernek meg kell tudnia jeleníteni egy felhasználó belépése során a belépés állapotát, amik a következők lehetnek: engedélyezett, tiltott és nem ismert kártya.
- A rendszernek rendelkeznie kell egy szolgáltatással, aminek segítségével valamilyen hálózati csatornán keresztül (pl.: HTTP kérések által) lekérdezhetőek és karbantarthatóak az RFID kulcsok, a felhasználók és a belépések.
- Gondoskodnunk kell arról, hogy ha egy felhasználó több alkalommal tiltott kártyával lép be, akkor üzenetet küldjünk a szolgáltatást üzemeltető rendszergazdának.

1.2. Nem funkcionális követelmények

- Egy Raspberry Pi 4 Model B eszköz még a projekt tervezése előtt beszerzésre került, ezért ez nem jelent külön kiadást, így a projekt ezzel kell megvalósítanunk, mert elégséges funkciókkal rendelkezik mind hardveres, mint szoftveres szinten.
- Egy bemutató eszköz elkészüléséhez az elektronikai elemeket tekintve maximum 10.000 forint összegű keretet szabunk meg. Ez egyben azt is meghatározza, hogy milyen hardverek jöhetnek szóba.

2. Tervezési fázis

2.1. Hardveres tervek

- A hardver elemek kiválasztása során figyelembe kell vennem, hogy a nem funkcionális követelményként meghatározott Raspberry Pi-hoz alkalmas eszközöket válasszak. Így ez esetben a GPIO lábkiosztása fogja nekünk megszabni azt, hogy pontosan milyen kommunikációs csatornán tudunk kommunikálni.
- Szükség esetén adott integrált áramköri panelek (pl.: RFID olvasó) lábait a csatlakozási pontokhoz kell forrasztani, ezért a forrasztóállomás beszerzéséről és egyéb kellékekről is gondoskodni kell.
- A bemutató eszközt próba panelen célszerű elkészíteni, mert az adott bekötés könnyen módosítható, illetve nem kell forrasztási műveleteket sem végezni ehhez.

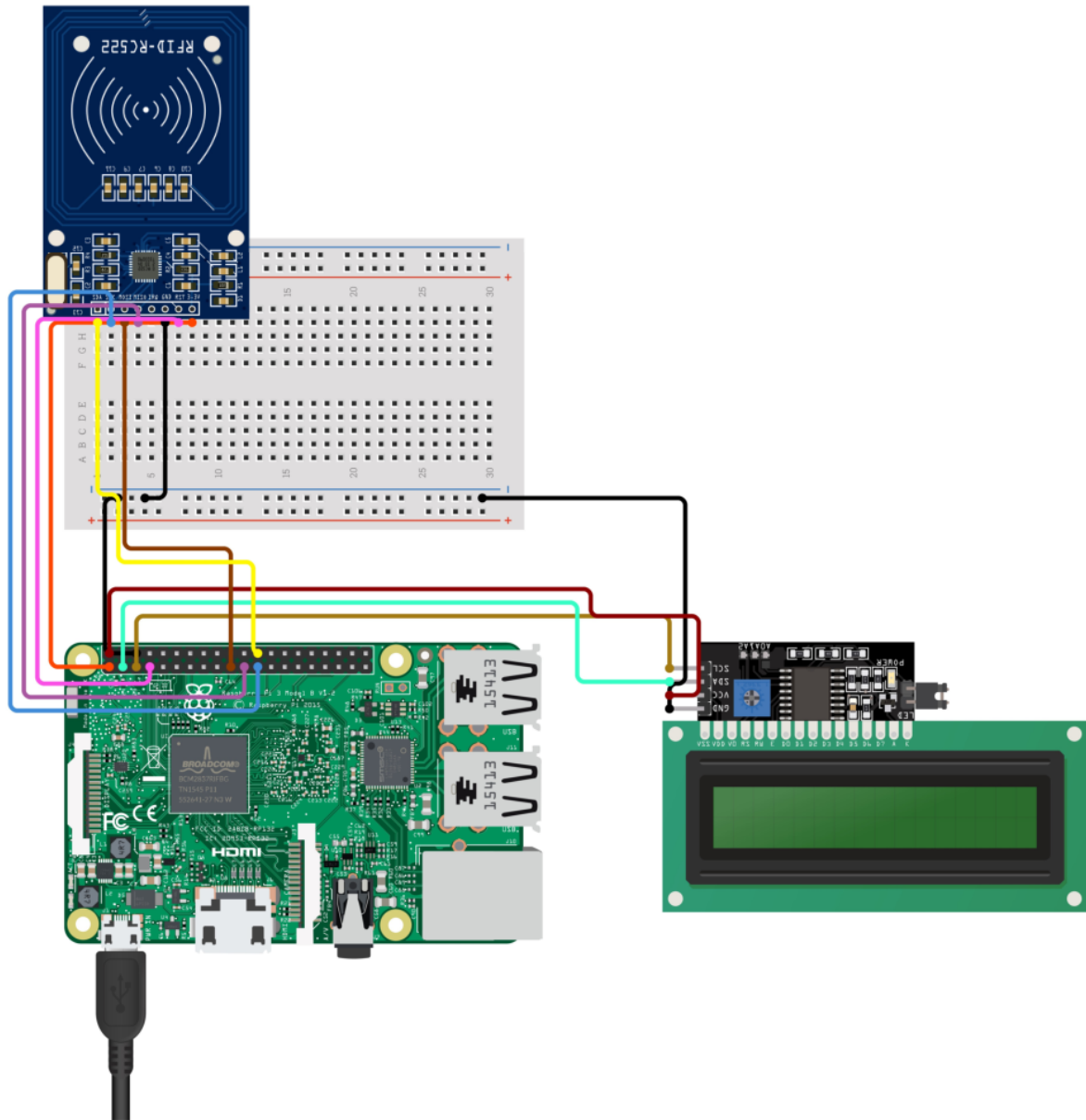
2.2. Szoftveres tervek

- A Raspberry Pi 4 Model B eszköz (lényegében egy ARM architektúrájú kis számítógépről van szó) rendelkezik a gyártó által fejlesztett operációs rendszerrel, ezért ennek használata tűnt a legcélszerűbbnek, mert könnyen kezelhető és széleskörű csomag támogatással rendelkezik.
- Követelményként szükségünk van egy adatokat szolgáltató alkalmazást fejleszteni, és mivel egy bemutató alkalmazásról van szó, ezért az egyszerűség kedvéért Python nyelven készül el, különböző *pip* csomagok használatával.

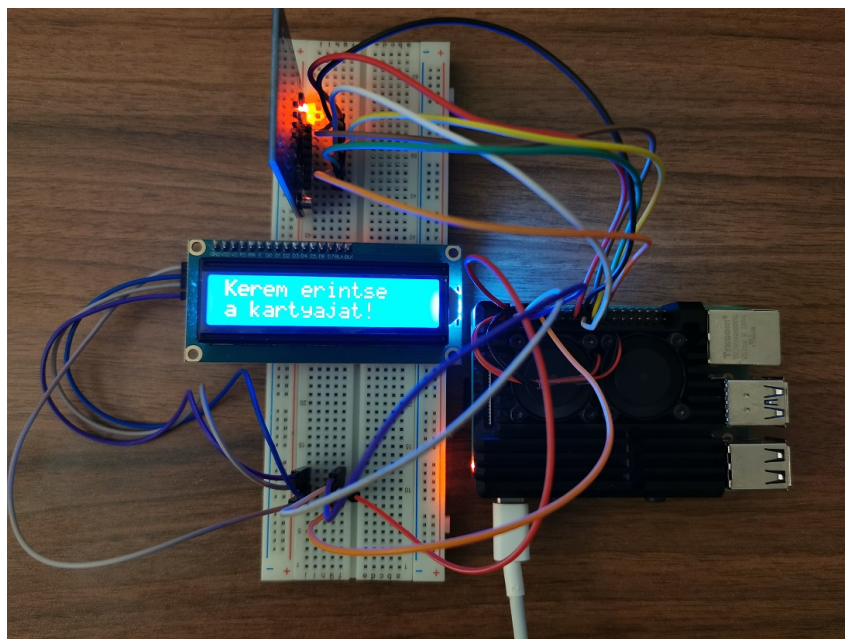
3. Hardver specifikáció

3.1. Próbapanel

A felhasznált próbapanel modellje: BB-102 (630/200)











































1. ábra. A hardveres terv próbapanelen



2. ábra. A projekt tényleges megvalósítása üzemkész állapotban

3.2. Raspberry Pi

Az eszköz rendelkezik egy 40 csatlakozási pontos GPIO interfésszel. Ennek van egy belső számozása, amit a Raspberry Pi rendszere tud kezelni, illetve egy 1-40-ig tartozó számsorozat, ami a teljes lábkiosztásra vonatkozik. A következő ábrán a GPIO# jelölés a belső számozást, a csatlakozási pontok melletti számok pedig a teljes lábkiosztást adják meg. Az LCD kijelző és az RFID olvasó a teljes lábkiosztás szerint van bekötve.

Raspberry Pi 4 Model B (J8 Header)					
GPIO#	NAME			NAME	GPIO#
	3.3 VDC Power	1		2	5.0 VDC Power
8	GPIO 8 SDA1 (I2C)	3		4	5.0 VDC Power
9	GPIO 9 SCL1 (I2C)	5		6	Ground
7	GPIO 7 GPCLK0	7		8	GPIO 15 TxD (UART)
	Ground	9		10	GPIO 16 RxD (UART)
0	GPIO 0	11		12	GPIO 1 PCM_CLK/PWM0
2	GPIO 2	13		14	Ground
3	GPIO 3	15		16	GPIO 4
	3.3 VDC Power	17		18	GPIO 5
12	GPIO 12 MOSI (SPI)	19		20	Ground
13	GPIO 13 MISO (SPI)	21		22	GPIO 6
14	GPIO 14 SCLK (SPI)	23		24	GPIO 10 CE0 (SPI)
	Ground	25		26	GPIO 11 CE1 (SPI)
30	SDA0 (I2C ID EEPROM)	27		28	SCL0 (I2C ID EEPROM)
21	GPIO 21 GPCLK1	29		30	Ground
22	GPIO 22 GPCLK2	31		32	GPIO 26 PWM0
23	GPIO 23 PWM1	33		34	Ground
24	GPIO 24 PCM_FS/PWM1	35		36	GPIO 27
25	GPIO 25	37		38	GPIO 28 PCM_DIN
	Ground	39		40	GPIO 29 PCM_DOUT
					
					
					
					
					
					
					
					
					
					
					
					
					
					
					
					
					
					
					
					

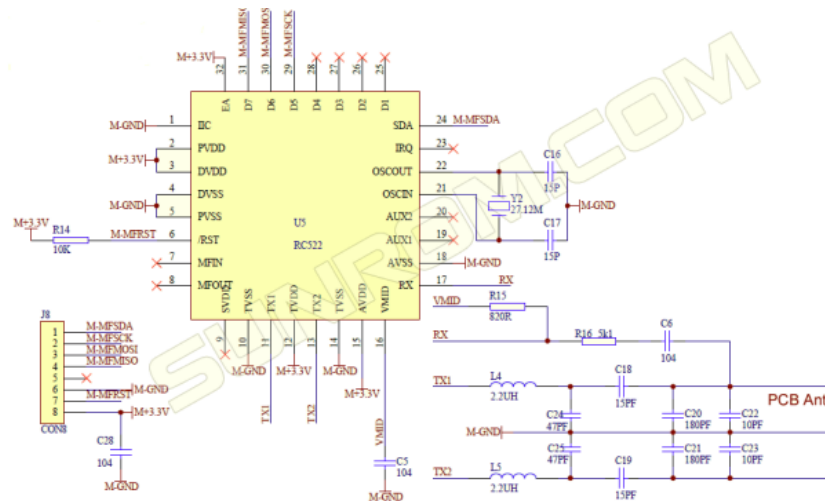
Attention! The GPIO pin numbering used in this diagram is intended for use with WiringPi / Pi4J. This pin numbering is not the raw Broadcom GPIO pin numbers.

<http://www.pi4j.com>

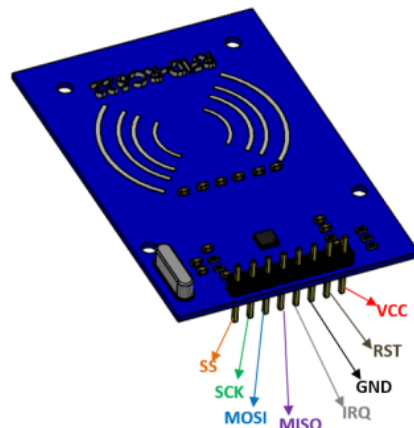
3. ábra. Raspberry Pi 4 Model B lábkiosztása

3.3. RFID olvasó

A felhasznált RFID olvasó modellje: RC522-MFRC. A kommunikáció az olvasóval SPI interfészen keresztül történik, melynek értékét ütemezett intervallumokban lekérdezzük.



4. ábra. Az RFID olvasó kapcsolási rajza



Pin Number	Pin Name	Description
1	VCC	Used to Power the module, typically 3.3V is used
2	RST	Reset pin – used to reset or power down the module
3	Ground	Connected to Ground of system
4	IRQ	Interrupt pin – used to wake up the module when a device comes into range
5	MISO/SCL/Tx	MISO pin when used for SPI communication, acts as SCL for I2c and Tx for UART.
6	MOSI	Master out slave in pin for SPI communication
7	SCK	Serial Clock pin – used to provide clock source
8	MOSI	Acts as Serial input (SS) for SPI communication, SDA for IIC and Rx during UART

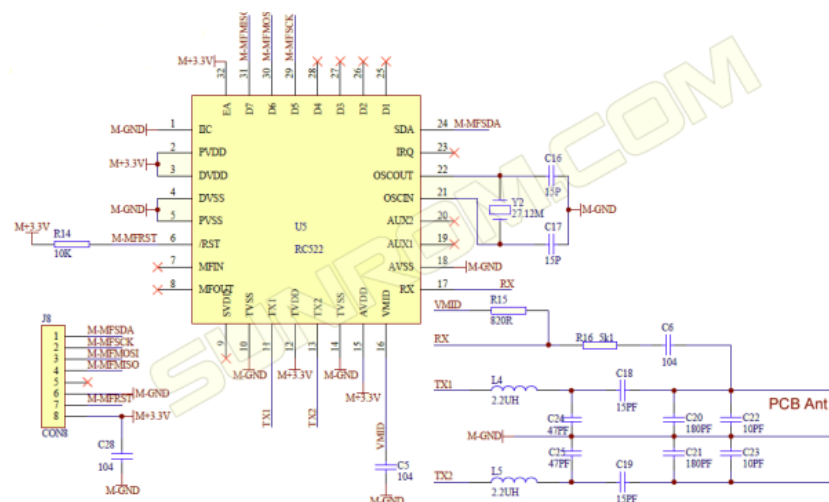
5. ábra. Az RFID olvasó lábkiosztása

RC522-MFRC olvasó bekötési terv		
Raspberry	RFID pin	
GPI0 pin	neve	
száma		
17	VCC	
22	RST	
20	GND	
21	MISO	
19	MOSI	
23	SCK	
24	SDA (MOSI)	

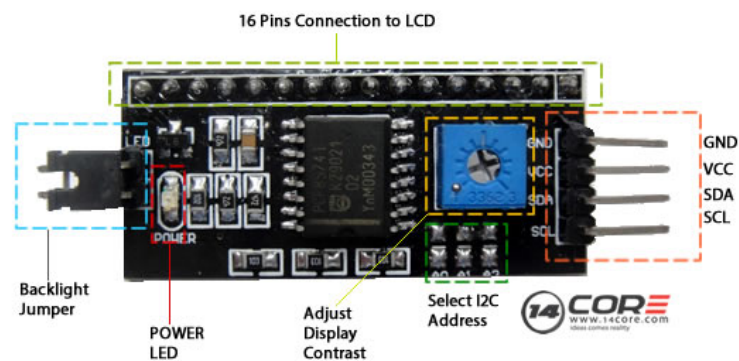
1. táblázat. Az RFID olvasó bekötési terve

3.4. LCD kijelző

A felhasznált LCD kijelző modellje: KC-1602-BB-I2C. A kommunikáció I²C csomagkapcsolt soros buszon keresztül történik.



6. ábra. Az LCD olvasó kapcsolási rajza



7. ábra. Az LCD kijelző kivezetési pontjai

KC-1602-BB-I2C lábkiosztás	
Pin	Funkció
GND	Földelési pont
VCC	5V tápellátás
SDA	Soros adatvonal
SCL	Soros órajel

2. táblázat. Az LCD kijelző felhasznált lábai

KC-1602-BB-I2C bekötési terv	
Raspberry	
GPI0 pin száma	RFID pin neve
9	GND
2	VCC
3	SDA
5	SCL

3. táblázat. Az LCD kijelző bekötési terve

4. Szoftver specifikáció

4.1. Raspberry Pi OS

A Raspberry Pi OS egy Debian alapú operációs rendszer, ami kifejezetten a Raspberry Pi hardverekhez lett létrehozva. A következő operáció rendszer van telepítve:

- Kiadási dátum: 2023 október 10.
- Rendszer: 64-bit
- Kernel verzió: 6.1
- Debian verzió: 12 (bookworm)

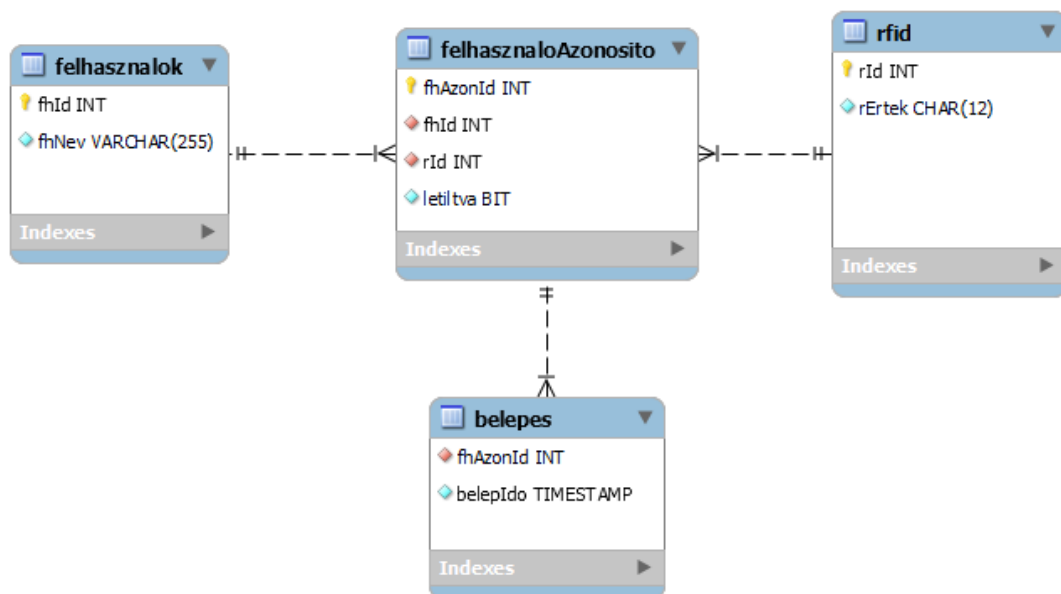
4.1.1. Docker

A Docker egy konténeres virtualizációs platform, lehetővé teszi a felhasználóknak, hogy szoftver-alkalmazásokat izolált környezetben futtassák. A konténereket könnyedén és gyorsan el lehet indítani, amik általában egy Dockerfile alapján vannak definiálva. A konténerben futott szoftver alapját az úgy nevezett *base image* adja meg. Ebben az esetben a MySQL adatbázis kezelő szoftverhez van használva, mert így el lehet kerülni a telepítési nehézségeket, valamint a konténert bármikor le tudjuk állítani vagy el tudjuk indítani. A konténer indítását a jelenlegi projekt esetén a szolgáltatáshoz tartozó indító szkript rendszerindításkor megteszi.

5. A fejlesztett kód részletezése

5.1. Adatbázis szerkezet

A szolgáltatás mögött egy MySQL adatbázis fut a korábban említett Docker konténerben, ami a belépési idők rögzítését, illetve az adott felhasználóhoz tartozó idők lekérdezését hívatott megvalósítani. Az adatbázis szerkezete a következő ábrán és táblázatokon van részletezve:



8. ábra. Egyed-kapcsolat modell

Felhasznalok			
Oszlop név	Típus	Tulajdonságok	Funkció
fhId	INT	UNIQUE NOT NULL AUTO_INCREMENT	A felhasználó azonosítója
fhNev	VARCHAR(255)	NOT NULL	A felhasználó neve

4. táblázat. Az felhasználókkal kapcsolatos alapinformációk tárolása

Rfid			
Oszlop név	Típus	Tulajdonságok	Funkció
rId	INT	UNIQUE NOT NULL AUTO_INCREMENT	Az RFID azonosítója
rErtek	CHAR(12)	UNIQUE NOT NULL	Az RFID kulcs értéke

5. táblázat. Az RFID kulccsal kapcsolatos alapinformációk tárolása

FelhasznaloAzonosito			
Oszlop név	Típus	Tulajdonságok	Funkció
fhAzonId	INT	UNIQUE NOT NULL AUTO_INCREMENT	Egy rekord azonosítója
fhId	INT	NOT NULL	A felhasznalok tábla fhId oszlopa
rId	CHAR(12)	UNIQUE NOT NULL	Az RFID tábla rId oszlopa
letiltva	BIT	NOT NULL DEFAULT 0	A felhasználóhoz rendelt RFID azonosító tiltásának állapota

6. táblázat. Egy felhasználó összekapcsolása több RFID kulccsal (egy-a-többhöz)

Belepes			
Oszlop név	Típus	Tulajdonságok	Funkció
fhAzonId	INT	NOT NULL	A felhasználóAzonosító fhAzonId oszlopa
belepIdo	TIMESTAMP	NOT NULL DEFAULT CURRENT_TIMESTAMP()	A belépés ideje
rId	CHAR(12)	UNIQUE NOT NULL	Az RFID tábla rId oszlopa
letiltva	BIT	NOT NULL DEFAULT 0	A felhasználóhoz rendelt RFID azonosító tiltásának állapota

7. táblázat. Egy felhasználóhoz tartozó RFID kulcs belépési idejének tárolása

5.2. Verziókövető rendszer

A projekt követelményeknek megfelelően a kód GitHubon van tárolva, ami a következő linken elérhető: https://github.com/mark182182/GKLB_INTM020_mikroelektromechanikai_rendszerek

5.3. Konfiguráció

Az alkalmazásban használt beállításokat egy [config.ini](#) fájlból olvassuk ki. A Pythonban képesek vagyunk a `configparser` csomag segítségével egy előre megadott struktúrában definiálni kulcs-érték párokat.

A felhasználásuk a következő módon történik:

- `[mysql]` alatt:
 - Host: A MySQL adatbázis IP címe
 - User: A MySQL adatbázishoz tartozó felhasználó
 - Password: A MySQL adatbázishoz tartozó jelszó
 - Database: A MySQL adatbázishoz neve
- `[smtp]` alatt:

- Host: A SMTP szerver címe
- Port: A SMTP szerverhez tartozó port
- User: A SMTP szerverhez szükséges felhasználó
- Password: A SMTP szerverhez szükséges jelszó
- FromAddress: Az emailt küldő címe
- ToAddress: Az emailt fogadó címe
- DebugLevel: Logikai változó, amely segítségével extra információt kapunk az SMTP szerverrel való kommunikációról

Dokumentáció a `configparser` csomagról [ezen a linken elérhető](#).

5.4. Indító szkript

Az implementáció a következő fájlban található: [start.sh](#)

5.5. SPI kommunikáció

Az implementáció a következő fájlban található: [rfid_spi.py](#)

5.6. I²C kommunikáció

Az implementáció a következő fájlban található: [lcd_2c.py](#)

5.7. Email küldés

Az implementáció a következő fájlban található: [smtp_client.py](#)

5.8. HTTP hívások

Baz