ТЕХНОЛОГИЧНО УЧИЛИЩЕ ЕЛЕКТРОННИ СИСТЕМИ
към ТЕХНИЧЕСКИ УНИВЕРСИТЕТ - СОФИЯ

ДИПЛОМНА РАБОТА

по професия код 481020 „Системен програмист“
специалност код 4810201 „Системно програмиране“

Тема: Създаване на умна ключалка за врата, отключваща се чрез QR код

Дипломант:
Михаил Цветанов Димитров

Дипломен ръководител:
маг. инж. Петър Стаменов

СОФИЯ

2 0 2 2

ТЕХНОЛОГИЧНО УЧИЛИЩЕ ЕЛЕКТРОННИ СИСТЕМИ
към ТЕХНИЧЕСКИ УНИВЕРСИТЕТ - СОФИЯ

Дата на заданието: 14.12.2021 г.
Дата на предаване: 14.03.2022 г.

Утвърждавам:.....
/проф. д-р инж. Т. Василева/

ЗАДАНИЕ
за дипломна работа

ДЪРЖАВЕН ИЗПИТ ЗА ПРИДОБИВАНЕ НА ТРЕТА СТЕПЕН НА ПРОФЕСИОНАЛНА КВАЛИФИКАЦИЯ
по професия код 481020 „Системен програмист“
специалност код 4810201 „Системно програмиране“

на ученика Михаил Цветанов Димитров от 12 а клас

1. Тема: Умна ключалка за врата, работеща с QR код

2. Изисквания:

- Да се избере механизъм за заключване и отключване на врата, който да е съвместим с едноплатков микрокомпютър.
- Да се изработи електронен модул, служещ за контролиране на електро-механичната част, на базата на едноплатков микрокомпютър и периферна електроника.
- Да се осигури автономно захранване и възможността за включване към електрическа мрежа.
- Да се разработи API, служещо за комуникация с мобилно приложение.

3. Съдържание
   3.1 Теоретична част
   3.2 Практическа част
   3.3 Приложение

Дипломант :.....
/ Михаил Цветанов Димитров /

Ръководител:.....
/ маг. инж. Петър Стаменов /

Директор:.....
/ доц. д-р инж. Ст. Стефанова /

ТЕХНОЛОГИЧНО УЧИЛИЩЕ ЕЛЕКТРОННИ СИСТЕМИ

СЪДЪРЖАНИЕ

Съдържане

Увод

Вградените микрокомпютърни системи се състоят от хардуерна и софтуерна част и са използвани на много места и в много различни сфери.

Вградените микрокомпютърни системи са създадени, за да се намали заетото пространство от електроника и да се разграничи функционалността на отделните компоненти. В последствие предназначението им се променя така, че да бъдат използвани и като самостоятелни устройства и системи.

Вградените микрокомпютърни системи се намират на много места из всекидневния живот, като почти всеки човек, имащ достъп до електричество, се е сблъсквал с такава. Из най-разпространените са перални за дрехи, съдомиялни, автоматизирани прахосмукачки, системи за наблюдение и устройства за заключване на врати. Последният вид система е избрана за тази дипломна работа.

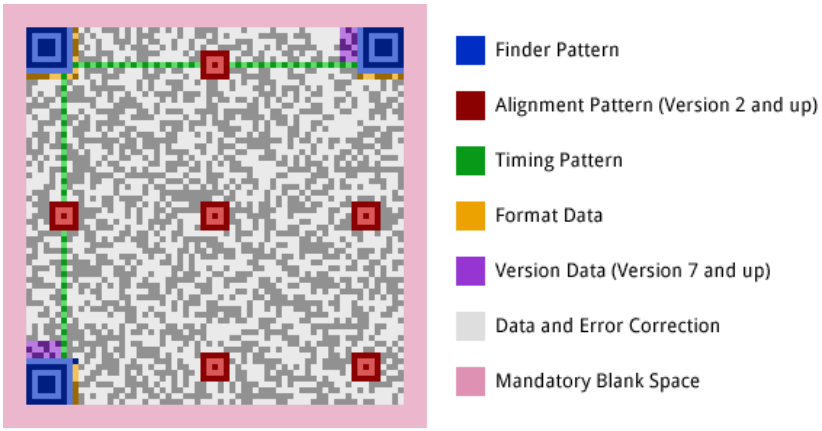
Целта на дипломният проект е проектирането и създаването на система за заключване на врати. Поставените функционалности са управление на заключващ механизъм чрез четене и обработка на QR кодове и възможност за настройване чрез Bluetooth.

ПЪРВА ГЛАВА

Проучване, подобни разработки и избрани части

1.1. Разчитане на информация и комуникация с външни устройства

Проектът се основава на управление на вградената микрокомпютърна система от мобилно устройство и получаване и обработване на информация, предоставена чрез QR код. QR(quick response) кодът представлява двуизмерен баркод, оптимизиран за четене от машини. Състои се от черни квадрати, поставени в квадратна форма, на бял фон. Един QR код съдържа 6 до 7 основни елемента, показани нагледно и с пример на фигура 1.1: ориентировъчна маркировка, подравняваща маркировка (за втора и по-висока версия QR код), оразмеряваща маркировка, информация за формата на записаните данни, информация за версията (за седма и по-висока версия QR код), записаните данни и маркировка, служеща за оправяне на грешки при четене, и задължително празно място.



Фигура 1.1 Елементи на един QR код [1]

Ориентировъчната маркировка се използва, за да може четецът да открие къде точно се намира QR кодът и да се разбере ориентацията му, за да се прочете информацията от останалите елементи.
Подравняващата маркировка е необходим при QR кодове от версия 2 нагоре поради по-големите размери на самия код и съответно информацията, която той съдържа. Служат за изправяне на

изображението ако се сканира код под ъгъл, защото иначе сканиращото устройство се затруднява с четенето и подреждането на информацията като двумерен масив.
Оразмеряващата маркировка служи за показване на размера на вписаната в QR кода информация.
Записаните данни в един QR код обикновено се форматира като се обръщат цветовете на определени сегменти от кода. В този елемент на кода се съхранява информацията за кои сегменти трябва да се обърнат при прочитане и обработване на данните.
В останалата шарка на един QR код се съдържат информация за версията му, данните, които трябва да се прочетат и обработят, и маркировка за поправяне на информацията при четене. Проверката за грешки и обработката им се случва по алгоритъма на Рийд-Соломон за оправяне на грешки[2] .

Комуникацията трябва да се осъществява чрез безжична мрежа, за което беше избрана технологията Bluetooth. Bluetooth е технология за късообхватна безжична връзка, изградена така, че да бъде с ниска консумация на енергия, поради което не е чак толкова бърза при пренос на информация. Намира приложение като заместител на кабелни мрежи с обхват до 10 метра и нисък трансфер на данни. Най-често се използва за комуникация при подвижни системи от устройства и между различни вградени микрокомпютърни системи.

1.2. Подобни проекти

Идеята не е нова и няма липса на разработки до момента. Има различни разработки на същата тема и с подобна функционалност. Най-популярните и разпространени подобни системи са LuckySystems QR Code Lock[3], QRLOCK[4] и Lock Changer[5].

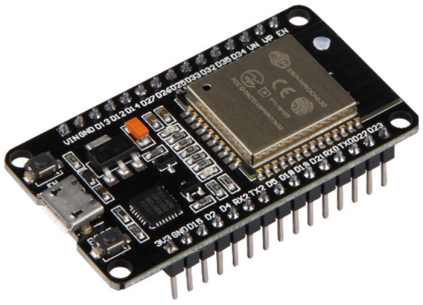
1.2.1. LuckySystems QR Code Lock – тази система е разработена от компанията LuckySystems и е предвидена като надграждане над обикновена умна ключалка, произведена от тях. Контролира се чрез приложение, което не е описано как се свързва със самото устройство.

1.2.2. QRLOCK – QRLOCK или INNOVATIVE QRLOCK са компания от Сингапур, работеща с QR кодове от много време по техни думи. Не може обаче да се намери техен продукт на пазара и няма изображения, демонстрации или документирана функционалност.

1.2.3. Lock Changer – Проект, разработен от белгийски студент по задание от университета. Представлява брава, разчитаща QR кодове и ползваща ги за ключове. Разликата с дипломната работа, описана в тази документация е начинът на настройване на устройството. За да се управлява Lock Changer трябва да се изгради и включи уебсайт и да се работи през него.

1.3. Избрани части

1.3.1. Микроконтролер: ESP32, показан на фигура 1.2, беше избран като най-подходящ за целта на тази разработка. Има необходимата памет и мощност да разчита и разкодира QR кодове. Има и специална дистрибуция разработена за работа с камера за умни устройства.

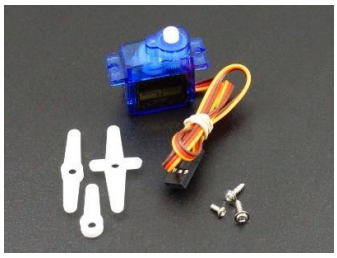


Фигура 1.2 ESP32CAM

1.3.2. Захранване: необходим е преобразувател, превръщащ електричество от електрическата мрежа към такова, което да е подходящо за вградени микрокомпютърни системи. Показан на фигура 1.3 е едноканалният модел RAC03-04SGB, защото е малък по размери и преобразува от 85 до 305 волта променлив ток към 5 волта постоянен ток.

Фигура 1.3 RAC04-05SGB Преобразувател

1.3.3. Заключващ механизъм: заключващият механизъм бе избран да се разработи вместо да се закупува, за да бъде лесна демонстрацията. Беше избран да се използва стъпков електромотор Servo Motor Micro SG90[6], показан на фигура 1.4, защото е малък по размери, лесен за управление и употреба и не се нуждае от много енергия.



Фигура 1.4 SG90 Micro тъпков електромотор

ВТОРА ГЛАВА

Основни функционалности на умна ключалка, работеща с QR код, схема на функционалността и избрани компоненти

2.1. Функционални изисквания към умна ключалка

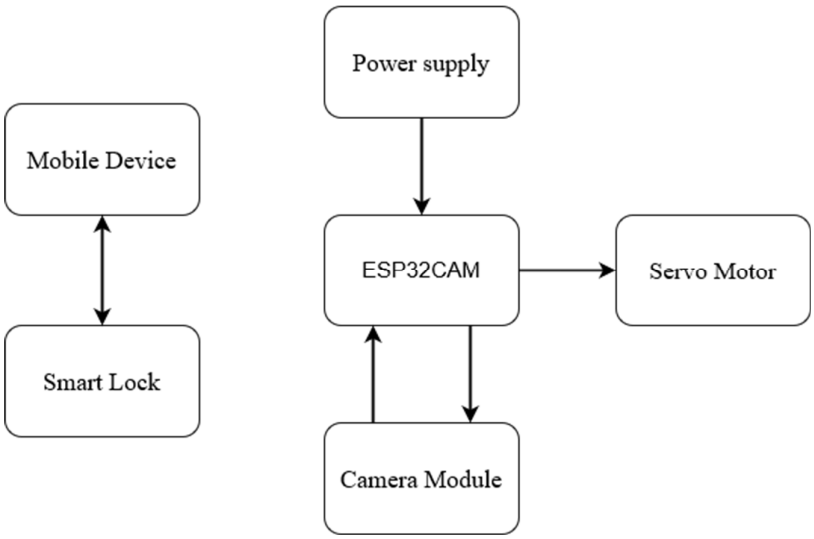
2.1.1. Проектът трябва да поддържа и използва разчитане и декодиране на QR кодове, служещи като ключове за отваряне.

2.1.2. Системата трябва да поддържа съхраняване и обработка на списък с QR ключове, които имат достъп до отключване на ключалката. Трябва да се осигури възможността за установяване на комуникация с външни устройства и правене на промени по списъка за достъп при прочетен фабричен ключ и подадена команда.

2.1.3. Устройството трябва да може да се включва в и захранва от електрическа мрежа, работеща с променлив ток при напрежение до 264V, на мястото, където ще бъде монтирано.

2.2. Блокова схема на устройството

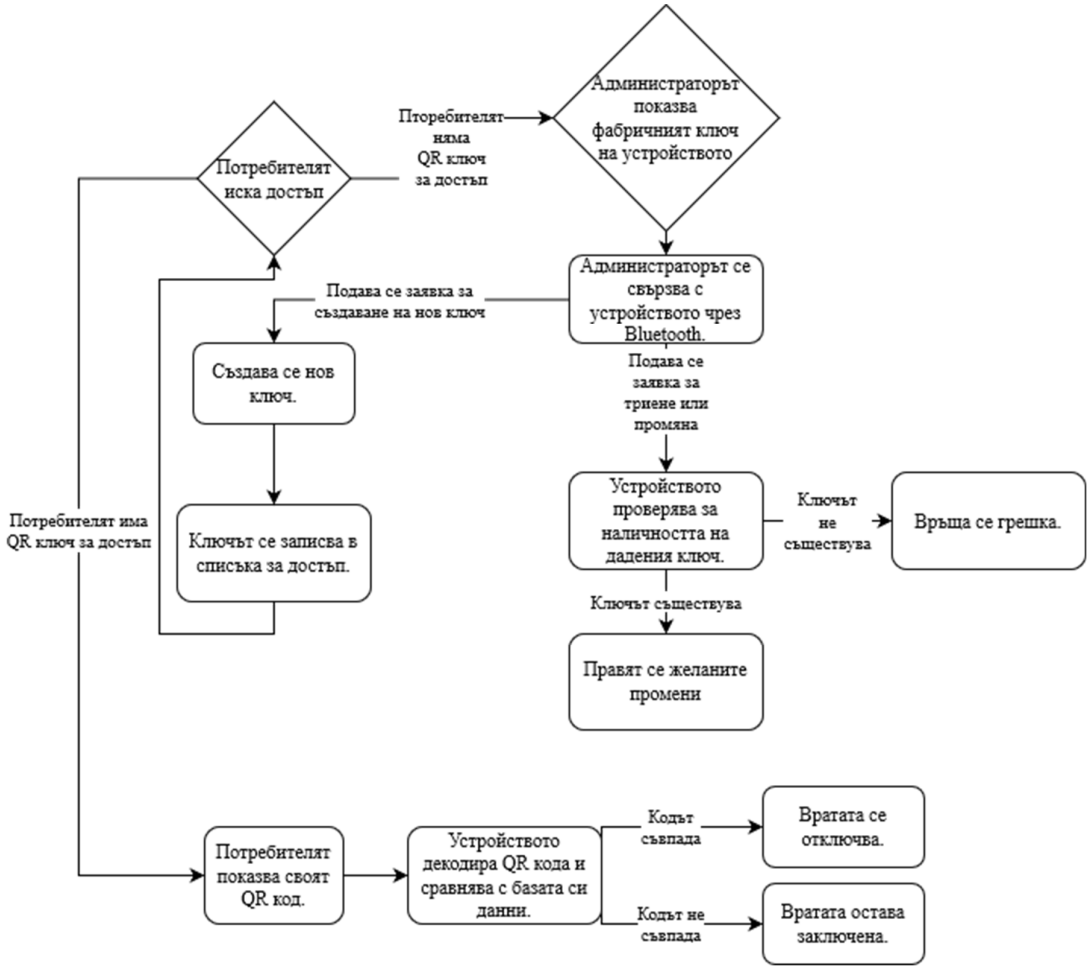
На фигура 2.1 е изобразена хардуерната блокова схема на устройството. На нея са представени графично взаимовръзките между различните елементи на системата.



Фигура 2.1 Блокова схема на устройството

Избраният микроконтролер бива захранван от избрания преобразувател. Камерата следи за QR кодове и подава информацията под формата на снимки към контролера, където те биват обработени и разчетени. Това се случва като на снимката се идентифицира QR кодът чрез маркерите, описани в първа глава, обработи се така, че да може записаните в него данни да бъдат временно прехвърлени към двумерен масив за съхранение докато се обработят и разчетат. Когато това приключи в микроконтролера се взема решение и се предприема действие, което, ако е отключване или заключване на ключалката, се подава като сигнал към стъпковия електромотор, служещ за отваряне и затваряне.

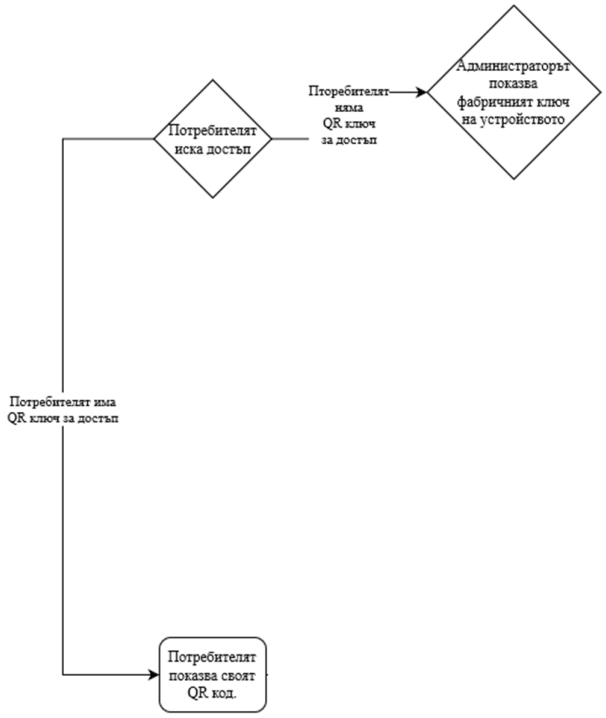
На фигура 2.2 е показана схема на предвиденият начин на употреба на устройството. В нея е описано взаимодействието между отделните



Фигура 2.2 Предвидена работа с устройството

елементи и събития. Работата със системата може да се раздели на няколко части: потребителска, администраторска, работа със списъка за достъп при добавяне и премахване на запис и вземане на решение от микроконтролера.

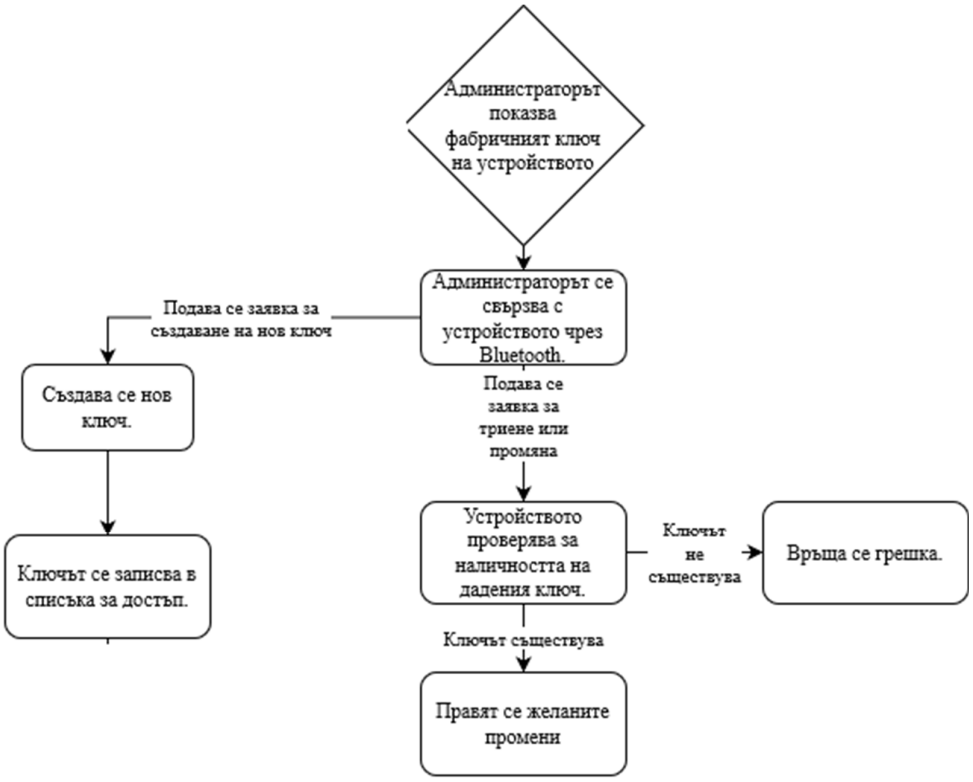
На фигура 2.3 се представя предвиденото потребителско въздействие върху устройството и системата. Когато иска да отключи ключалката, обикновеният потребител трябва да представи своят QR код, служещ като ключ. Ако той притежава такъв, трябва да го покаже на камерата от устройството. Ако няма ключ и желае достъп то потребителят може да се свърже с администратор.



Фигура 2.3 Потребителски действия

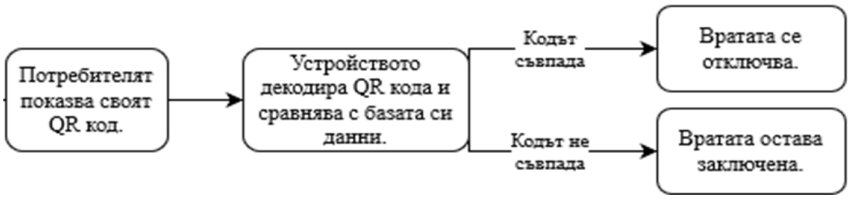
На фигура 2.4 е извадена частта от схемата, отнасяща се до въздействието на администратор върху системата. Когато е необходимо администраторът може да използва фабрично зададеният главен ключ, като по този начин включва Bluetooth комуникацията с устройството и го

поставя в режим на обработка и промяна списъка за достъп. От там може да се подадат заявки за създаване на ключ или премахване на съществуващ такъв. При заявка за създаване трябва да се подадат и параметри, които, ако е успешно записването, биват вкарани в списъка и превърнати в запис, който ще се проверява при бъдещи опити за влизане. Ако се избере изтриване на ключ се очаква да се подаде кой от записите да бъде премахнат. Когато това се направи се извършва проверка дали желаният ключ съществува. Ако да той се изтрива от списъка и в бъдеще няма да се сверява с него при опити за влизане. Ако не съществува такъв запис се връща съобщение, че такъв ключ не съществува.



Фигура 2.4 Работа на администратора с устройството

На фигура 2.5 е показано нагледно решението дали да се отвори ключалката при показване на ключ. При показване на QR ключ микроконтролера разчита информацията от него и проверява дали тя съвпада с някой от записите в списъка си за достъп. Ако съвпада ключалката се отключва за 10 секунди, след което автоматично се заключва. Ако не съвпада с никой от записите ключалката остава заключена



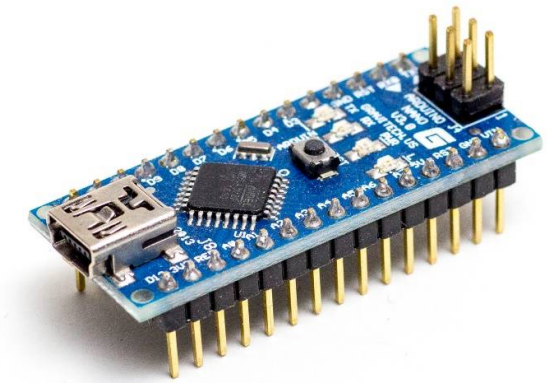
Фигура 2.5 Отваряне на ключалката

2.3. Съпоставка на компоненти

На пазара има много различни варианти за възможни компоненти и части за вградени микрокомпютърни устройства. Основните компоненти, необходими за този проект са: микроконтролер за управление на системата; основен захранващ модул, който е трансформатор от променлив ток към прав ток с напрежение, зависещо от необходимото за функциониране на микроконтролерът; стъпоков електромотор, служещ за заключващ механизъм. Понеже заключващият механизъм е предвиден само като демонстрация в този проект, то той ще бъде опростен и избран така, че да може да се използва с минимални ресурси. Най-популярният и достъпен стъпков мотор на пазара, който да функционира на достатъчно ниски напрежения че да не изисква отделно захранване, е Servo Motor Micro SG90. Поради това той ще бъде избран като най-добър за необходимостите на тази дипломна работа.

2.3.1. Съпоставка на компоненти (микроконтролери)

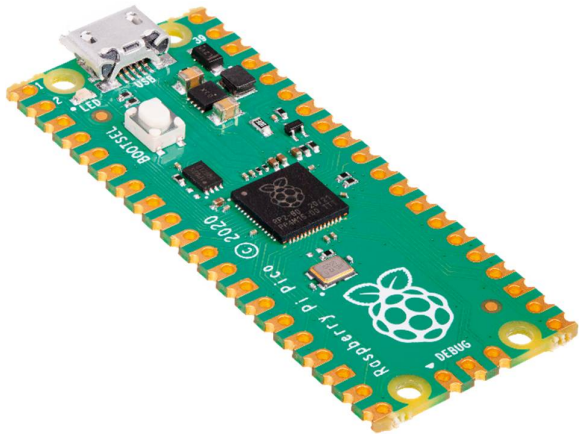
При избирането на компоненти бе направено проучване и сравнение между различни възможности за всяка от необходимите части, за да се намери най-добрият и подходящ вариант за устройството. Микроконтролерът служи за управление на стъпковия електромотор и разчитане и декодиране на QR код, което изисква немалко ресурси. Трябва да има и достатъчно място за съхранение на списъка с ключове, имащи достъп. Понеже устройството ще се включва в постоянно работеща електрическа мрежа консумацията на енергия не е от значение. Бяха разгледани 3 варианта: контролери с Arduino Nano, ESP32CAM и Raspberry Pi Pico RP2040.



Фигура 2.6 Arduino Nano

2.3.1.1. Първо ще бъде разгледан вариантът Arduino Nano, показан на фигура 2.6. Представлява малък по размери микроконтролер, работещ на Arduino рамката за писане и компилиране на код. Платката работи при входно напрежение от минимум 7V до максимум 12V. Има 30 извода за вход и изход, 8 от които могат да се ползват за вход на аналогови сигнали. Има малко оперативна памет в сравнение с другите избори. Няма вградена поддръжка на Bluetooth и Wi-Fi и трябва да се намерят външни модули за да

бъде възможна комуникацията между системата и външни устройства. Не може да се добавя външна памет и има малко пространство за съхранение на информация. Няма вграден вход за камера и трябва да се намери камера, съвместима с Arduino Nano и да се осигури захранване за камерата. Този вариант е най-скъпият в сравнение с останалите.



Фигура 2.7 Raspberry Pi Pico RP2040

2.3.1.2. Разглежда се Raspberry Pi Pico RP2040, показан на фигура 2.7. Работи при напрежение до 3.3V, но има вграден преобразувател от 5V към 3.3V. Има 26 извода за вход и изход, 4 от които могат да се ползват а вход на аналогови сигнали. Има достатъчно оперативна памет, за да може със сигурност да поддържа четене и разкодиране на QR кодове и управление на стъпков електромотор. Има достатъчно налично място за съхранение на данни за списъка за достъп, но ако се добавят още функционалности в бъдеще е възможно да бъде недостатъчно. Няма възможността за поставяне на външна памет. Няма вграден Bluetooth или Wi-Fi модул и трябва да се намери съвместим такъв. Няма вграден вход за камера, което значи че ще трябва да се търси съвместима такава и

да се осигури захранване. Изходното напрежение, с което микроконтролера може да захранва външни елементи е максимално 3.3V, което е недостатъчно за голяма част от стъпковите електромотори и е вероятно да трябва да се осигури отделно захранване за заключващия механизъм. Този микроконтролер е най-евтин от трите варианта.

2.3.1.3. Накрая ще бъде разгледан и представен вариантът ESP32CAM, вече показан в първа глава на фигура 1.2. ESP32CAM е микроконтролер, работещ при напрежение от 5V до 12V. Има 9 извода за вход и изход, от които нито един не може да се ползва за аналогов вход. Има най-много оперативна памет от всички варианти. Няма вътрешна памет, но може да се постави външно хранилище под формата на SD карта, на която да се съхранява информация. Има вградени модули за Bluetooth 4.2 и BLE стандартите и Wi-Fi комуникация. Има вграден вход и фабрична поддръжка за камера. Този вариант е между останалите два по цена.

На таблица 2.8 е представена съпоставка между основните свойства на разгледаните микроконтролери. С оглед на спецификите на различните контролери бе избран ESP32CAM като най-добър вариант. Налични са Bluetooth и Wi-Fi модули за комуникация и има вградена поддръжка на камера. Цената не е проблемна, защото в нея се включва и микрокамера модел OV2640. Това води до по-ниска цена за снабдяване с всички необходими основни части от колкото би била цената при другите варианти.

Таблица 2.8 Съпоставка на микроконтролери

	Arduino Nano	Raspberry Pi Pico	ESP32CAM
Входно напрежение	7V до 12V	3.3V до 5V	5V до 12V
Брой I/O изводи	30	26	9
Оперативна памет	2KB	264KB	520KB
Вътрешно хранилище	32KB	2MB	8MB
Външно хранилище	Не	Не	Да
Вграден Bluetooth	Не	Не	Да
Вграден Wi-Fi	Не	Не	Да
Вграден вход за камера	Не	Не	Да
Изходно напрежение	3.3V и 5V	3.3V и 5V	3.3V и 5V
Размер	18 x 45 mm	51 x 21 mm	27 x 40.5 mm
Поддържани езици	Arduino, C, C++	Arduino, C, C++, Python	Arduino, C, C++, Python
Цена	18$	10$	14$

2.3.2. Съпоставка на компоненти (захранващи блокове)

На таблица 2.9 е показана съпоставка между двата най-популярни варианта за захранване на вградени микрокомпютърни системи, предвидени да се включват директно в електрическата мрежа. Разгледаните варианти за трансформатори са RAC04-05SGB[7], произведен от Recom Power, и TMLM 04103[8], произведен от Traco Power. RAC-05SK преобразувателят е по-толерантен при почти всички параметри, има по-висок обхват на поддържано входно напрежение и по-голям диапазон на температурата при която работи. Малко по-скъп е от

TMLM 04103, но е по-устойчив и е наличен в България. С оглед на тези разлики RAC04-05SGB бе избран за по-надеждният вариант.

Таблица 2.9 Съпоставка на преобразуватели

	RAC04-05SGB	TMLM 04103
Входно напрежение (променлив ток)	85V до 305V	90V до 264V
Изходно напрежение (постоянен ток)	5V	5V
Температура на работа (°C)	-40 до +85	-25 до +60
Изходен ток	800 mA	800 mA
Размери (mm)	37 * 24 * 15	36.5 * 27 * 17.1
Наличност в България	Да	Не
Цена	19.80 лв.	31.01 лв.

2.3.3. Съпоставка на комуникационни методи

След избора на микроконтролер, който има възможността за комуникация чрез Bluetooth и различните негови стандарти и Wi-Fi, трябва да се реши и коя ще е по-удобната технология за комуникация.

Таблица 2.10 Основна съпоставка на комуникационните варианти

	Bluetooth	BLE	Wi-Fi
Скорост	Висока	Ниска	Много висока
Енергийна консумация	Ниска	Много ниска	Висока
Обхват	Къс	Къс	Дълъг

Както е представено на таблица 2.10, различните комуникационни технологии и протоколи си имат плюсове и минуси. Wi-Fi е бърз, с голям обхват и позволява свързването на множество устройства в една обща мрежа, обаче консумацията на енергия също е висока. При BLE (Bluetooth Low Energy) имаме много ниска консумация на енергия, но обхватът е малък и скоростта също е ниска. Освен това се ограничава броят на устройства, които могат да са свързани едновременно. При стандартен Bluetooth (версия 4.2) енергийната консумация е по-ниска от тази на Wi-Fi, но е по-висока от BLE. Обхватът е къс като при BLE, но за сметка на това скоростта е по-висока, но не по-висока от тази на Wi-Fi. При Bluetooth също се ограничава броят на едновременно свързаните устройства. Този проект е предвиден да се използва от близо, да има достатъчно бърз пренос на данни, да може да се захранва от един захранващ модул и да не може да се свързва повече от едно устройство. С оглед на това Bluetooth е най-добрият избор за комуникация, понеже покрива всичките изисквания.

ТРЕТА ГЛАВА

Подробно описание на елементната база

3.1. Микроконтролер

ESP32CAM е един от най-популярните микроконтролери за вградени микрокомпютърни системи, работещи с камера. Той е относително евтин, с много функции и добри показатели. На таблица 3.1 са изведени основните му характеристики.

Таблица 3.1 Характеристики на ESP32CAM

	ESP32CAM
Необходимо захранване	5V
Температура на работа	-20°C до 85°C
Максимална консумация	310mA на 5V
Цифрови изводи	9
Оперативна памет	520KB
Вътрешно хранилище	8MB
Размери	27 * 40.5 * 4.5 mm

ESP32CAM върши идеална работа за този проект, защото е много устойчив на външни условия като температура и вибрации. Има достатъчно ниска консумация, за да не се налага да се търси по-силно и съответно по-скъпо и по-обемно захранване. Има изключително бърз процесор в сравнение с останалите продукти. Има достатъчно оперативна памет, за да може без проблеми да сканира QR кодове и да изпълнява и други функции. Има много вътрешна памет за разлика от голяма част от останалите контролери на пазара, а може и да се сложи външно

хранилище под формата на MicroSD карта памет, която да е и лесна за работа и интеграция. Има вграден извод за камера OV2640 или OV7670. Поддържа и Bluetooth и Wi-Fi комуникация и качване на изображения от камерата през съответната комуникация. Има малко на брой изводи но голяма част от тях поддържат различни начини за комуникация и предаване на сигнали като пулсово широчинна модулация, асинхронна и синхронна серийна комуникация и серийна периферна комуникация. Микроконтролерът не заема много обем и е компактен, което го прави перфектен за употреба при устройства, които трябва да са мощни, но да не заемат твърде много място. Тази дипломна работа е точно такъв проект. Освен това заради разнообразието на характеристиките на ESP32CAM то предоставя най-голям потенциал за бъдещо развитие на проекта от всички микроконтролери. Възможно е да се добавят още елементи, процеси и логики, а ако свърши вътрешното хранилище има външно такова, което може да се ползва както за резервно така и за основно.

3.2. Камера

OV2640 е камерата, която идва в комплект заедно с ESP32CAM микроконтролерът. Като отделен елемент камерата работи при напрежение до 3.3V и консумира до 125mA в активно състояние и работи безпроблемно при температури от -40 до 95 градуса по Целзий. Разделителната ѝ способност е по-висока от колкото на споменатата в предната подточка алтернатива. Поддържа споделяне на живо и записване на видеа в качество 1600 x 1200 пиксела с 15 кадъра в секунда. Поддържа и записването на кратки видеа или изображения под формат GIF или JPEG, което не е често срещано при камери във вградени микрокомпютърни системи. Поради възможностите за развитие на проекта предоставени от микроконтролера, тази камера е най-добрият избор за тази разработка. Ако се реши да се използва за видео наблюдение освен за скенер на QR кодове.

3.3. Стъпков електромотор

Избраният стъпков електромотор TowerPro SG-90 върши отлична работа за предназначението му в този проект. Работи при напрежение до 5V, консумира най-много до 360mA ток. Функционира при температури от 0 до 55 градуса по Целзий. Полезно е за тази разработка, защото не изисква твърде много енергия, има 180-градусово въртене, което е идеално за заключващ механизъм и е сравнително силно и издръжливо в сравнение с останалите стъпкови електромотори с подобен размер и енергийно изискване.

3.4. Захранващ модул

Захранващият модул бе избран да е RAC04-05SBG, произведен от Recom Power. На таблица 3.2 са дадени неговите основни характеристики.

Таблица 3.2 Характеристики на захранващ модул

BASIC CHARACTERISTICS						
Parameter	**Condition**			**Min.**	**Typ.**	**Max.**
Internal Input Filter						Pi-type
Input Voltage Range [3,4]	nom. Vin = 230VDC			85VAC 120VDC		305VAC 430VDC
Input Current	115VAC 230VAC				85mA 55mA	
Inrush Current	cold start at 25°C	115VAC 230VAC				10A 20A
No load Power Consumption						75mW
Input Frequency Range	AC Input			45Hz		65Hz
Minimum Load				0%		
Power Factor	115VAC 230VAC				0.55 0.42	
Start-up Time	115VAC, 230VAC				30ms	1s
Hold-up time	115VAC 230VAC				10ms 40ms	
Internal Operating Frequency	100% load at nominal Vin				65kHz	
Output Ripple and Noise [9]	20MHz BW	0°C to 85 °C	3.3Vout 5Vout 9Vout 12Vout 15Vout 24Vout			100mVp-p 100mVp-p 120mVp-p 150mVp-p 200mVp-p 240mVp-p
		-30 °C to 0 °C	3.3Vout 5Vout 9Vout 12Vout 15Vout 24Vout			200mVp-p 200mVp-p 250mVp-p 250mVp-p 300mVp-p 300mVp-p

Този трансформатор бе избран поради относително компактните си размери и издръжливостта си. Много е устойчив на външни влияния като температура, вибрации и електрически шокове, както е показано на таблица 3.3. Работи при най-разпространените напрежения за електрически мрежи по света и има пренебрежима загуба на енергия при преобразуване на електричеството.

Таблица 3.3 Външни влияния и фактори

ENVIRONMENTAL			
Parameter	**Condition**		**Value**
Operating Temperature Range	@ natural convection 0.1m/s	full load	-40°C to + 70°C
		refer to ***"Derating Graph"***	-40°C to + 85°C
Maximum Case Temperature			+100°C
Temperature Coefficient			0.03%/K
Operating Altitude			3000m
Operating Humidity	non-condensing		5% - 95% RH
Pollution Degree			PD2
Shock			20G/11ms pulse, 3 times at each x, y, z axes
Vibration			10-150Hz, 2G 10min./1cycle, period 60min. along x,y,z axes for 6 cycles
Design Lifetime	+25°C +50°C		90 x 10^3 hours 62 x 10^3 hours
MTBF	according to MIL-HDBK-217F, G.B.	+25°C +50°C	>900 x 10^3 hours >198 x 10^3 hours

ЧЕТВЪРТА ГЛАВА

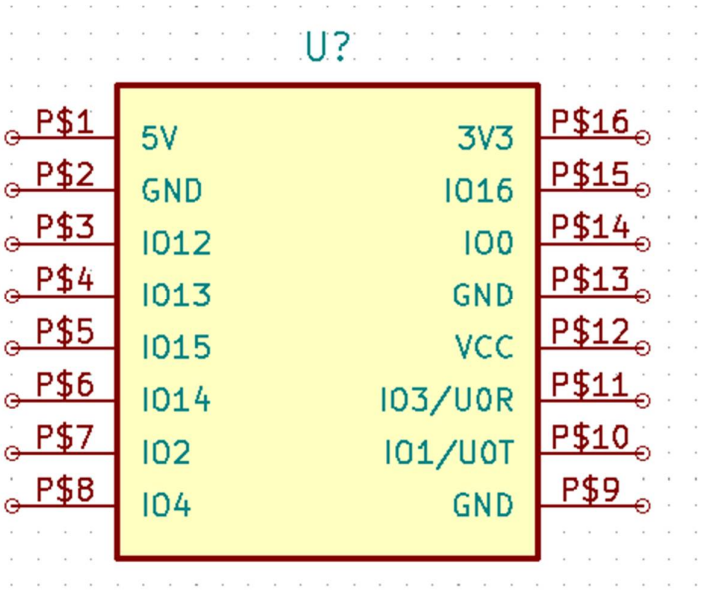
Проектиране на електрическа схема и печатна платка

4.1. Електрическа схема

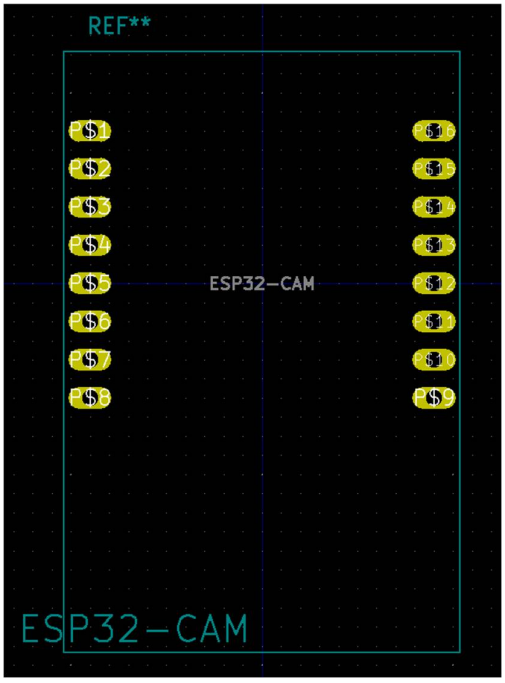
За изграждането на електрическата схема беше използвана платформата KiCad. Тя беше избрана поради по-ниско ниво на трудност при употреба в сравнение с останалите подобни програми.

За да се изгради електрическата схема беше необходимо да се намерят елементите и техните означение и ако се налага да се правят промени по тях. Ще бъдат разгледани избраните елементи, техните схемни означения и графични изображения.

На фигура 4.1 е показан схемното означение на ESP32CAM, а на фигура 4.2 е показано графичното.



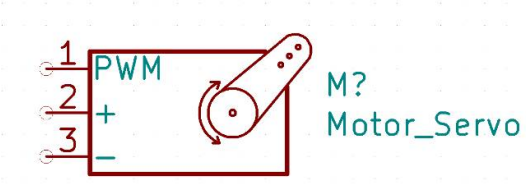
Фигура 4.1 Схемно означение на ESP32CAM



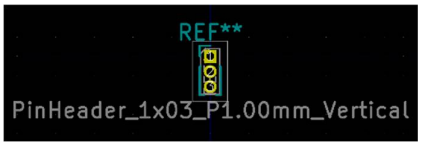
Фигура 4.2 Графично изображение на ESP32

Схемното и графичното изображения за микроконтролера липсват от KiCad и бяха импортирани като потребителска библиотека[10].

На фигури 4.4 и 4.5 ще бъдат показани съответно схемното и графичното изображение на стъпковия електромотор. Първоначално този елемент нямаше графично изображение, но му беше избрана through-hole връзка с 3 извода, разположени вертикално.

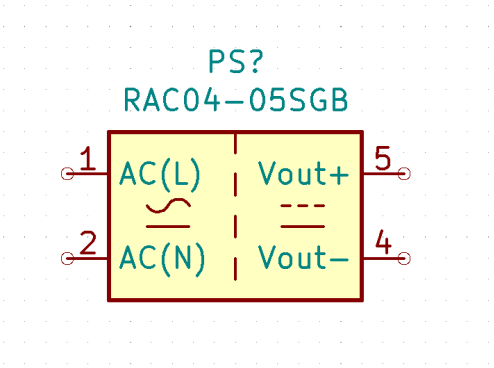


Фигура 4.4 Схемно означение на стъпков електромотор

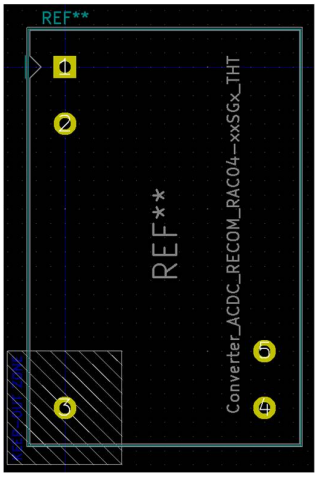


Фигура 4.5 Избрано графично изображение на стъпков електромотор

На фигури 4.6 и 4.7 са съответно изобразени схемното означение и графичното изображение на избраният захранващ модул.

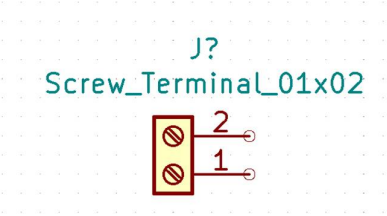


Фигура 4.6 Схемно означение на захранващ модул

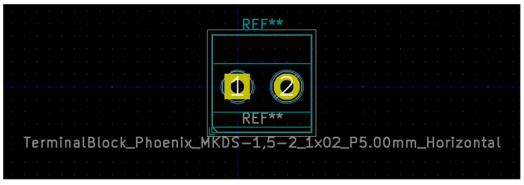


Фигура 4.7 Графично икображение на захранващ модул

За да може захранващият модул да се включва в електрическата мрежа както е предвиден трябваше да се избере някаква свръзка. Беше избрана винтова клема с два терминала, чиито схемен символ и графично изображение са показани съответно на фигури 4.7 и 4.8

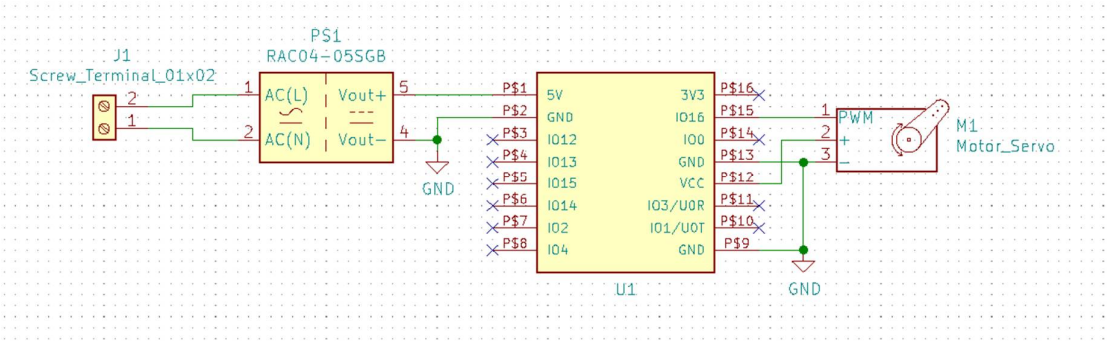


Фигура 4.7 Схемно изображение на винтова клема



Фигура 4.8 Графично изображение на винтова клема

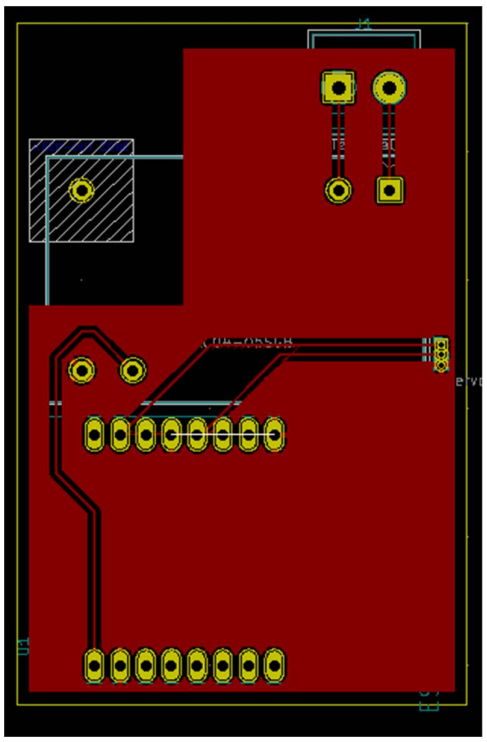
На фигура 4.9 е показана завършената електрическа схема на устройството.



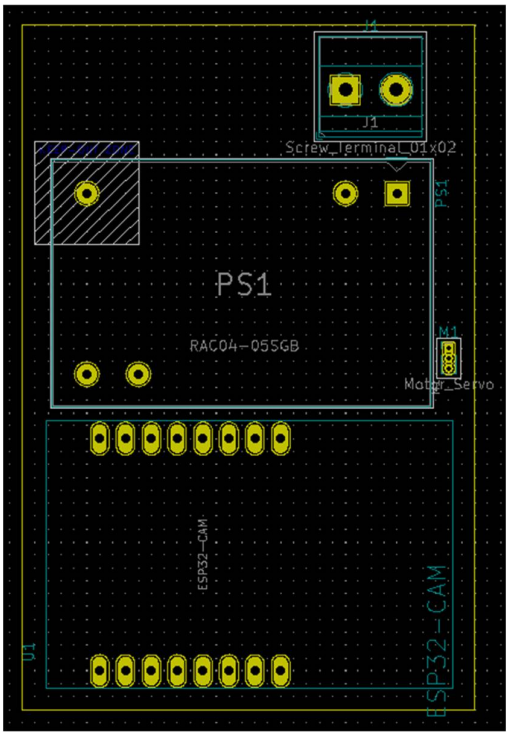
Фигура 4.9 Завършена електрическа схема

4.2. Печатна платка

За проектирането на печатната платка също е използвана програмата KiCad. Проектирана е еднослойна платка с размери 44.45mm * 67.31mm. Използва се само един размер писта от 0.250mm ширина. От консумациите на стъпковия електромотор и на микроконтролера, описани в трета глава, и от тях може да се пресметне че максималният ток на цялата система е приблизително 670 mA. Чрез калкулатор се пресмята, че при дебелина на пистата от 0.1mm необходимата ширина е 0,157mm.
На фигура 4.10 е показан графичният оригинал на платката със слой спокйа, а на фигура 4.11 е показан само слой елементи.



Фигура 4.10 Графичен оригинал на печатна платка



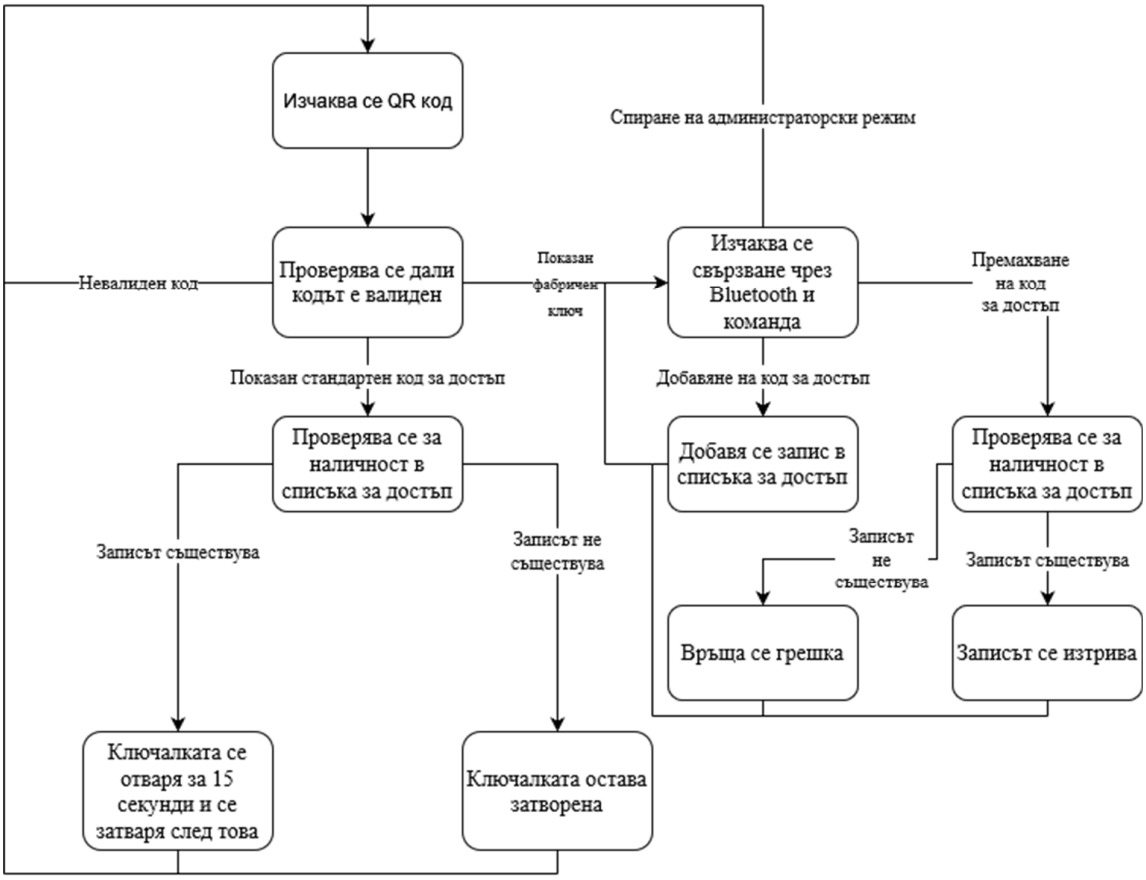
Фигура 4.11 Слой елементи

ПЕТА ГЛАВА

Създаване на алгоритъм и софтуер за управление на умна ключалка, работеща с QR код

5.1. Алгоритъм на работа

За да се контролира устройството е изграден алгоритъм, чиято нагледна схема е показана на фигура 5.1.



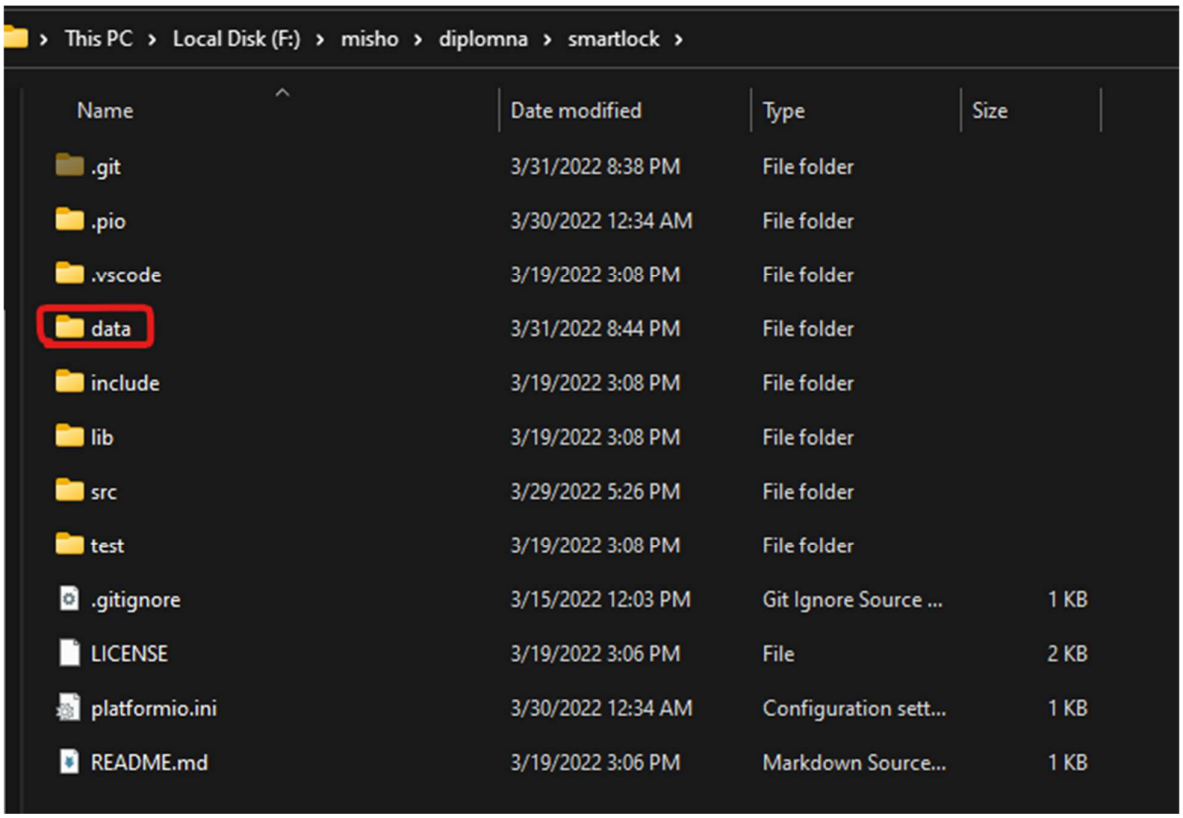
Фигура 5.1 Алгоритъм на работа на устройство

Устройството изчаква да се покаже QR код. През това време не се изпълнява нищо друго освен да търси гореспоменатия код. Когато се представи код устройството оперира на няколко стъпки. Първо се проверява дали подаденият QR код е валиден и дали може да се разчете. Ако кодът е невалиден или повреден се връща грешка и започва изчакването за код наново. Ако кодът може да бъде разчетен се проверява

дали е фабричният администраторски ключ или не. Ако не е се проверява в списъците за достъп, за да се реши дали вратата да се отвори. Ако информацията от кода се намери в списъците, то ключалката се отваря и се затваря автоматично след 15 секунди. Ако не е намерен ключалката остава затворена. Ако кодът съвпадне с фабричният ключ се включва Bluetooth и се изчаква връзка. Когато се осъществи връзката се изчакват инструкции от устройството, с което е осъществена връзка. Инструкциите могат да бъдат добавяне или изтриване на ключ от списъка за достъп, който се съхранява на самото устройство. Ако се подаде заявка за изтриване се проверява в списъка дали съществува такъв запис. Ако да то той се трие от списъка и устройството се връща на изчакване за команди. Ако не то се връща грешка и устройството се връща на изчакване за команди. Ако се подаде заявка за спиране то устройството изключва Bluetooth и се връща към изчакване на QR кодове за сканиране.

5.2. Реализация на работа с файлове

За да може да се съхранява и обработва списъкът за достъп трябва да има направено четене и обработване на файлове. За постигането на това бе избрана да се използва библиотеката SPIFFS[11], която обслужва файлове. Файлът се съхранява на вътрешното хранилище на микроконтролера, но може да се премести и на външна памет, която трябва само да се инициализира. Първо трябва да се направи отделна папка и файл, от който да се чете в нея. Папката е необходимо да се казва „data“ и да е поставена в главната папка на проекта, както е показано на фигура 5.2, за да може контролерът да я зачита като хранилище за информация.



Фигура 5.2 Необходимата папка „data“ в главната папка на проекта

След като папката е поставена на създадена на предвиденото място и е достъпна за системата се преминава към кода. Тук ще се разгледа само отварянето и затварянето на файлове, защото останалите използвани функции са важни за други сегменти от кода. На фигура 5.3 е показано отварянето за употреба и зареждането на един файл чрез SPIFFS.

```
File authList = SPIFFS.open("/access_list.txt", "r");
if(!authList)
{
  Serial.println("There was an error opening the access list\n");
  return;
}
```

Фигура 5.3 Отваряне на файл

На първият ред от показаният код се създава променлива от специфичният за SPIFFS тип File, която приема стойността на показател към отворен

файл. Това позволява достъп до него за четене и писане спрямо настройките. Понеже това е в най-общата част от функцията за сканиране на QR кодове и не се изисква презаписване файла се отваря в режим на четене. Това се случва изцяло чрез метода open(), която приема като параметри път към даден файл и режим на обработка, който в случая е „r“. След това се прави проверка дали файла е отворен успешно и дали може да се обработва и разчита. Ако всичко е наред устройството продължава работата си докато ако има грешка цялостната функция прекратява работата си.

След като файлът е отворен той трябва да се преведе в улеснен за обработка вид. За целта е избрано информацията да се вземе от списъка за достъп и да се сложи в списък от тип vector, който представлява масив от даден вид сложни елементи. Това е показано на фигура 5.4.

```
vector<String> authVect;
while(authList.available())
{
  authVect.push_back(authList.readStringUntil('\n'));
}
authList.close();
```

Фигура 5.3 прехвърляне на информация в Vector, съдържащ символни низове

Създава се променлива vector, съдържаща символни низове. След това в цикъл се чете от вече отвореният файл и се чете до нов ред. Когато се стигне до нов ред, който е избран за демонстративен разделител на ключовете от списъка, последното прочетено, което в случая включва целия ред, бива записано като последен елемент на vector-а. По този начин ние симулираме файла, съдържащ списъците за достъп, без реално да го държим отворен и да рискуваме повреда по него. Също така данните стават много по-лесни за обработка като елементи на един голям масив. След като приключим презаписването на файловете затваряме файла по вече споменатите причини.

5.3. Четене на QR кодове и комуникация

Основната задача на този проект е да обработва и реагира на QR кодове. Това е постигнато чрез употреба на библиотеката ESP32QRCodeReader[12], която позволява намиране на гореспоменатите кодове, четенето и декодирането им и обработка на информацията от тях. За да се използва камерата като скенер първо трябва да се създаде като обект, който да може да бъде достъпван от останалата част от програмата.

```
ESP32QRCodeReader scanner(CAMERA_MODEL_AI_THINKER);
```

Фигура 5.4 Създаване на обекта скенер

На фигура 5.4 е показано как точно се създава обектът четец на QR кодове. Това е обект идващ от гореспоменатата библиотека и като стойност приема моделът на микроконтролера, който ще се използва за разкодиране на кодове. След това трябва да се създаде структура от тип произхождащ от същата библиотека, която ще съдържа разчетената информация от прочетения QR код, както е показано на фигура 5.5.

```
struct QRCodeData qrCode;
```

Фигура 5.5 Структура за съхранение на информацията от кода

След като са изпълнени необходимите приготовления се пристъпва към самото намиране и съхранение на информация.

```
if(scanner.receiveQrCode(&qrCode, 100))
{
  if(qrCode.valid)
  {
    if(strcmp((char*)qrCode.payload, authVect[0].c_str()) == 0)…
    else …
  }
  else
  {
    Serial.println("Invalid data\n");
  }
}
```

Фигура 5.6 Сканиране и проверка за валидност

На фигура 5.6 е представен процесът на сканиране за QR код и неговата проверка. Извиква се функцията на вече създадения скенер за прочитане на код. receiveQrCode() работи като взема настоящия кадър и го записва в подадена променлива като първи аргумент, след което камерата не заснема нови кадри за време, посочено като втори аргумент на функцията, като то се брои в милисекунди. За да функционира правилно трябва подадената променлива да бъде адресът на структурата, използвана за съхранение на информацията от сканирания QR код. След като се получи и запише информацията тя трябва да се провери, което се случва с вградена променлива на структурата. Ако информацията е невалидна се връща грешка, а ако е валидна се продължава с действия, според това което има в кода.

След като се потвърди, че информацията е използваема, се продължава с обработката ѝ и вземането на решения спрямо нея. Както се вижда на фигура 5.6 са предвидени 2 възможности ако разчетените данни са валидни. В първия случай се сравнява получената информация с фабричният администраторски ключ, който за улеснение се намира на първи ред в списъка за достъп, но може да се изнесе в напълно отделен файл без да има проблеми. Във втория случай не се засича администраторския ключ и се продължава както е предвидено за обикновен потребител.

5.3.1. Засечен администраторски код

Когато се засече фабричният администраторски ключ устройството влиза в режим за настройване чрез Bluetooth. За да бъде възможно това трябва първо да се създаде BluetoothSerial обект от вградената Bluetooth библиотека в ESP32CAM, служещ за свързване и комуникация, както е показано на фигура 5.7. След това трябва да се стартира самият процес по свързване и комуникация, както е показано на фигура 5.8. В случая устройството ще бъде откриваемо под име „Smartlock“, както е зададено в параметъра на метода за включване.

```
BluetoothSerial lockBT;
```

Фигура 5.7 Инициализация на Bluetooth Serial обект

```
lockBT.begin("SmartLock");
```

Фигура 5.8 Включване на Bluetooth модул

След като се включи Bluetooth модула се отваря наново списъка за достъп аналогично на фигура 5.3, с разликата, че този път се отваря в режим на четене и писане, вместо само на четене. Това се постига като се смени вторият параметър в метода за отваряне на „FILE_WRITE“. След като успешно се отвори списъкът за достъп програмата влиза в режим на изчакване на инструкции през вече отвореният Bluetooth канал.

```
while(true)
{
  String btData = lockBT.readString();
  if(strcmp("add", btData.c_str()) == 0)···
  else if(strcmp("delete", btData.c_str()) == 0)···
  else if(strcmp("stop", btData.c_str()) == 0)···
}
```

Фигура 5.9 Поддържани инструкции

На фигура 5.9 е показано вземането на информация през установената Bluetooth комуникация и трите поддържани команди. За да се разчете информацията, получена през отвореният комуникационен канал с външно устройство, се прави променлива, в която да се съхраняват данните. Поради начинът на получаване на информация чрез Bluetooth може да се използват методи, за да се чете информацията като различни типове. За този проект е най-удобно да се използва методът readString(), който автоматично превръща получената информация към символен низ, каквито се използват често в тази разработка. За момента се поддържат само 3 операции: за добавяне на запис, изчистване на запис и спиране на администраторския режим. Разбира

се коя от операциите ще се изпълнява като се сравни полученият символен низ със съответните ключови думи, както е показано на гореспоменатата фигура.

5.3.1.1. Заявка за добавяне

Ако подаденият символен низ съвпадне с необходимият за добавяне на още записи програмата влиза в режим на изчакване на ключове, под формата на символни низове, които да се запишат на края на отвореният файл.

```
if(strcmp("add", btData.c_str()) == 0)
{
  while(true)
  {
    String newAuthData = lockBT.readString();
    if(strcmp("stop", newAuthData.c_str()) == 0)
    {
      lockBT.println("Exited addition mode");
      break;
    }
    else if(newAuthData.length() > 0)
    {
      adminAccessList.seek(SeekEnd);
      adminAccessList.println(newAuthData);
      authCount++;
    }
  }
}
```

Фигура 5.10 Добавяне на запис

На фигура 5.10 е показан методът на добавяне на нов запис в списъка за достъп. Когато се подаде командата „add“ програмата влиза в безкраен цикъл, който на всяка итерация сканира за нова информация изпратена по комуникационния канал. Ако се получи командата „stop“ то цикълът прекъсва, изпраща се съобщение обратно по Bluetooth комуникацията и устройството спира да се опитва да добави нови записи в списъка за достъп. Ако се получи символен низ с дължина по-голяма от 0 то тя ще се запише на края на отворения файл. Това се случва като първо намираме края на

файла и след това записваме получената през Bluetooth информация на нов ред в него. Накрая увеличаваме общият брой на записите в файла.

5.3.1.2. Заявка за изтриване

При получен символен низ, съвпадащ с необходимия за влизане в режим на изтриване на записи, програмата се включва в безкраен цикъл, в който се проверява за получени данни през комуникационния канал на всяка итерация, и се опитва да изтрие ключ от списъка за достъп, който да съвпада с подадените данни. На фигура 5.11 са представени подробно възможните действия при получена заявка за изтриване на запис. Още при подаването на заявката за влизане в режим на изтриване се дефинират две променливи, които да сочат дали програмата е била спряна и дали търсеният запис е бил открит. Аналогично на метода за добавяне на записи се дефинира променлива от тип символен низ, която да съхранява получената през Bluetooth информация. Отново аналогично на метода за добавяне има взети мерки да не се четат само празни символни низове и се поддържа команда за прекратяване на тази функционалност при заявката „stop“. Влиза се в цикъл, в който се проверява дали има получена информация по комуникационния канал. Ако има такава започва проверка под формата на друг цикъл с брояч и се следи дали броячът е стигнал до стойността на броя на записите. Ако стойностите им съвпадат се отбелязва в съответната променлива, че такъв запис не е намерен. В проверяващия цикъл се гледа за две неща: дали функцията трябва да спре и дали търсеният символен низ съвпада със стойността на vector-а от ключове, към която сочи броячът. Ако се е подал сигналът да се прекрати функцията това се записва в съответната променлива и проверителният цикъл бива прекъснат. Ако търсеният запис съвпадне с елемента на позиция,

съвпадаща с текущата стойност на брояча, в списъка, започва процесът по премахване на търсеният запис.

```
else if(strcmp("delete", btData.c_str()) == 0)
{
  while(true)
  {
    bool stopped = false;
    bool found = true;

    String delAuthData = lockBT.readString();
    if(delAuthData.length() != 0)
    {
      for(int i = 0; i < authCount; i++)
      {
        if(strcmp("stop", delAuthData.c_str()) == 0)
        {
          stopped = true;
          break;
        }
        else if(strcmp(delAuthData.c_str(), authVect[i].c_str()) == 0)
        {
          adminAccessList.seek(SeekSet);
          uint8_t delSize = authVect[i].length() + 1;
          for(int x= 0; x < authCount; x++)
          {
            String testDel = adminAccessList.readStringUntil('\n');
            if(strcmp(authVect[i].c_str(), testDel.c_str()) == 0)
            {
              adminAccessList.seek(-testDel.length(), SeekCur);
              break;
            }
          }
          char rem[delSize + 1];
          for(uint8_t x = 0; i < (delSize - 1); i++)
          {
            rem[x] = ' ';
          }
          rem[delSize - 1] = '\r';
          rem[delSize] = '\n';
          adminAccessList.print(rem);
          authCount--;
          lockBT.println("Successfuly deleted from access list");
          break;
        }
        if(i == authCount - 1)
        {
          found = false;
        }
      }

      if(found == false)
      {
        lockBT.println("No such entry found");
      }
      if(stopped == true)
      {
        lockBT.println("Exited deletion mode");
        break;
      }
    }
  }
}
```

Фигура 5.11 Метод за премахване на записи

Първо се взема дължината на записа и се добавя 1 към стойността, заради скрити символи от символния низ, които не се показват по принцип. След това се включва цикъл, в който се сравняват записа от vector-а, който е съвпаднал с желания запис за триене, и отделни редове от файла. Когато се намери съответствие между някой от редовете на списъка за достъп и елемента от vector-а файловия дескриптор в списъка се премества на началното на новия ред който е съвпаднал. След това се създава нова променлива, която да съдържа толкова символа като оригиналната, но да е празна. След като се създаде променливата тя се презаписва на мястото на записа, който е желан да бъде изтрит. Така записът спира да съществува и няма да се пропуска, докато не се добави отново. След това се изпраща съобщение, че записът е успешно заличен и се прекъсва проверителният цикъл. Накрая се прави проверка на променливите за спиране и намерен запис. Ако променливата за намерен запис е променена на „невярно“ се изпраща съобщение през Bluetooth, че такъв запис не е намерен. Ако променливата за прекъсната функция е променена на „вярна“ се изпраща съобщение по комуникационния канал, че бива прекъсната функцията за изтриване на записи и се излиза от съответния цикъл.

5.3.1.3. Заявка за прекратяване на работата

При получена заявка за прекратяване на работата, както е показано на фигура 5.12, се изпраща съобщение по комуникационния канал, че се изключва администраторският режим, затваря се списъкът за достъп, отворен за четене и писане, изключва се комуникационният канал през Bluetooth и се рестартира функцията за сканиране на QR кодове.

```
else if(strcmp("stop", btData.c_str()) == 0)
{
  lockBT.println("Exiting admin mode\n");
  adminAccessList.close();
  lockBT.end();
  return;
}
```

Фигура 5.12 Прекратяване на административен режим

5.3.2. Засечен потребителски код

Ако не се засече фабричният администраторски ключ, но QR кодът е валиден, се влиза в режим на обслужване на обикновени потребители.

```
else
{
  int i = 0;
  while(true)
  {
    if(i = authCount)
    {
      Serial.println("Access denied, no such key\n");
      break;
    }
    if(strcmp((const char*)qrCode.payload, authVect[i].c_str()) == 0)
    {
      Serial.println("QR key accepted, opening lock\n");
      openLock();
      delay(10000);
      closeLock();
      break;
    }
    i++;
  }
}
```

Фигура 5.13 Стандартно обслужване

На фигура 5.13 е показан начинът на операция, когато кодът е валиден, но не е администраторският. Създава се брояч и се започва цикъл, в който при всяка итерация се проверява дали броячът е със същата стойност като броя на записите и ако да се излиза от цикъла и започва да се търси QR код отново. Ако се намери съвпадение между

предоставения QR код и информацията в него и запис от списъка за достъп ключалката се отключва, както е показано по-долу, седи отключена за 10 секунди и се заключва, след което цикъла се прекъсва и започва да се сканира за QR код наново.

Самият стъпков електромотор се контролира чрез функциите, включени от библиотеката ESP32Servo[13]. Първо трябва електромоторът да се инициализира като достъпен обект (фигура 5.14), след което да се обяви на кой извод ще бъде прикачен (фигура 5.15), за да може да се управлява и накрая да му се зададе позиция, на която да застане съответно при заключване и отключване (фигури 5.16 и 5.17).

```
Servo lockServo;
```

Фигура 5.14

Създаване на серво обект

```
lockServo.attach(LOCK_PIN);
```

Фигура 5.15

Прикачване на серво към извод

```
void closeLock()
{
  lockServo.write(0);
}
```

Фигура 5.16

Заключване на ключалка

```
void openLock()
{
  lockServo.write(180);
}
```

Фигура 5.17

Отключване на ключалка

ЗАКЛЮЧЕНИЕ

В тази дипломна работа беше представена разработката на умна ключалка, работеща с QR кодове. За целта беше направено подробно проучване и сравнение на различни електрически елементи и хардуерни реализации. Бяха подробно описани целите и задачите на проекта и устройството беше проектирано така, че да ги удовлетворява. Бяха изградени електрическа схема и печатна платка на базата на подбрани най-добри за предвидената си работа елементи. Беше планиран и написан алгоритъм за управление на проектираната система, представен чрез няколко схеми.

Най-големи трудности бяха:

- Работа с KiCad софтуера, поради липса на опит при работа с подобни платформи.
- Почти пълна липса на информация за сходни разработки, независимо дали са комерсиални или лични проекти.
- Забавено доставяне на необходимите части.

Бъдещи възможности за развитие биха могли да включват:

- Добавяне на още функционалности при разчитане на QR код.
- Създаване и споделяне на QR код от устройството.
- Имплементация на комуникация чрез Wi-Fi.
- Свързване със специално разработено мобилно приложение, позволяващо контрол над устройството.

ИЗПОЛЗВАНА ЛИТЕРАТУРА И ИЗТОЧНИЦИ

1. Поредица от статии, разглеждащи работата на QR кодовете, тяхната функционалност и кои с изграждащите им компоненти - https://www.matchadesign.com/news/blog/qr-code-demystified-part-1/
2. Алгоритъм на Рийд-Соломон за корекция на сгрешена информация в матрица - https://www.cs.cmu.edu/~guyb/realworld/reedsolomon/reed_solomon_codes.html
3. http://luckysystems.us/product/qr-code-lock/
4. https://www.iqrlock.net/
5. https://www.instructables.com/LockChanger/
6. TowerPro Micro Servo 90 datasheet - https://components101.com/sites/default/files/component_datasheet/SG90%20Servo%20Motor%20Datasheet.pdf
7. Recom Power RAC04-05SGB Datasheet - https://eu.mouser.com/datasheet/2/468/RAC04_GB-1711323.pdf
8. Traco Power TMLM04103 Datasheet - https://www.mouser.bg/datasheet/2/687/tmlm-519430.pdf
9. Подробна електрическа схема и описание на ESP32CAM от AI Thinker - https://randomnerdtutorials.com/esp32-cam-ai-thinker-pinout/
10. Схемно и графично изображение на ESP32CAM - https://www.snapeda.com/parts/ESP32-CAM/AI-Thinker/view-part/
11. SPIFFS файлова система за ESP32 - https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/storage/spiffs.html
12. Библиотеката за четене на QR кодове чрез ESP32CAM - https://github.com/alvarowolfx/ESP32QRCodeReader
13. Библиотека за управление на стъпкови електромотори, изработена по подобие на оригиналната Arduino библиотека - https://registry.platformio.org/libraries/madhephaestus/ESP32Servo