  
**ТЕХНИЧЕСКИ УНИВЕРСИТЕТ - СОФИЯ**

**Факултет компютърни системи и технологии**

**Дипломна работа на тема**:

**СИСТЕМА ЗА АВТОМАТИЗИРАНО УПРАВЛЕНИЕ НА ПАРКИНГ**

|  |  |
| --- | --- |
| **София 2024** | **Дипломант:**  **Калоян Каменов Борисов КСИ, 121220159**  **Дипломен ръководител:**  **Ас. Иво Гергов** |

Съдържание

[**Списък с фигури 4**](#_Toc167811414)

[**Увод 5**](#_Toc167811415)

[**1. Постановка на дипломната работа, цели и задачи. 6**](#_Toc167811416)

[**1.1 ПРЕДИСТОРИЯ И МОТИВАЦИЯ 6**](#_Toc167811417)

[**1.1.1 Предистория 6**](#_Toc167811418)

[**1.1.2 Мотивация 8**](#_Toc167811419)

[**1.1.3 Потенциални препядствия 9**](#_Toc167811420)

[**1.2 Objectives and Goals 9**](#_Toc167811421)

[**1.3 Scope and Limitations 10**](#_Toc167811422)

[**2. Функционално описание. 11**](#_Toc167811423)

[**2.1 System Overview 11**](#_Toc167811424)

[**2.2 Key Components and Their Functions 11**](#_Toc167811425)

[**2.3 Communication Protocol 11**](#_Toc167811426)

[**2.4 Data Collection and Processing 11**](#_Toc167811427)

[**2.5 User Interface 11**](#_Toc167811428)

[**2.6 Use Cases 11**](#_Toc167811429)

[**3. Програмна реализация. 12**](#_Toc167811430)

[**3.1 комуникационна архитектура 12**](#_Toc167811431)

[**3.1.1 Комуникационна архитектура във възловата мрежа 12**](#_Toc167811432)

[**3.1.2 Комуникационна архитектура между потребителско устройство и възловата мрежа 12**](#_Toc167811433)

[**3.1.3 Комуникационна архитектура между възли за изчисляване на разстояние 12**](#_Toc167811434)

[**3.2 Системна архитектура на потребителското приложение 12**](#_Toc167811435)

[**3.3 Системна архитектура на възел 12**](#_Toc167811436)

[**4. Ръководство за използване и примери за употреба.  13**](#_Toc167811437)

[**4.1 Примерна постановка при наличие на Wi-Fi мрежа 13**](#_Toc167811438)

[**4.2 Примерна постановка без Wi-Fi мрежа 13**](#_Toc167811439)

[**4.3 Настройване на системата 13**](#_Toc167811440)

[**5. Заключение 14**](#_Toc167811441)

[**Съкращения 15**](#_Toc167811442)

[**Източници 16**](#_Toc167811443)

# Списък с фигури

[1 Layer Arcitecture 7](#_Toc167128811)

# Увод

Паркингът е част от нашето ежедневие, която е позната на всеки собственик на моторно превозно средство. Това е място, предназначено за временно пребиваване на моторното превозно средство (МПС). Паркингите обикновено не представляват особено технологична структура. Нормалният паркинг представлява голямо, отворено място, където да могат да се паркират МПС удобно. Обикновено има охранител, който се грижи превозните средства да са защитени от недоброжелатели. Охранителят също следи влизащите и излизащите автомобили, като също играе ролята на касиер, при когото се плаща престоят на МПС-то в паркинга.

Тази система може да работи при по-малки паркинги, където е лесно да се следят влизащите и излизащите автомобили. При тези по-малки паркинги, също не се създава натоварено движение на входа или изхода на паркинга. Не се случва често да е необходимо да се изчакват коли. В днешно време, много хора предпочитат да използват лични превозни средства, като автомобили, за да се придвижват от едно място до друго. Това създава растяща нужда от повече паркинги. Заради ограниченото място, не е добра опоция да се изграждат големи количества паркинги, а вместо това да има по-малко, но по-големи паркинги. Тези по-големи паркинги създават трудности при плащане и организация на вход и изход. Тук идва нужда за изграждане на система, която да позволява лесен достъп до паркинга и лесен изход от него. Забавянето при влизане или излизане трябва да бъде сведено до минимум.

За да се наблюдават превозните средства, които влизат и излизат от паркинга, е необходимо да се следят уникалните МПС. Това става лесно, тъй като всяко МПС има уникален регистрационен номер, който е поставен на видно място върху превозното средство. Интелигентната паркинг система може да използва система от камери, която да наблюдава автомобилите и да проверявва регистрационните им номера в база данни от регистрирани потребители и така да ограничава достъпът на неоторизирани лица. Използвайки тази система за достъп е възможно да разберем точният момент на вход и на изход на дадено МПС и да ги запишем в сървър. Така можем да позволим потребителите да плащат сметките си онлайн, без да имаме нужда от касиер на паркинга, който да бави процеса на изход от паркинга.

# Постановка на дипломната работа, цели и задачи.

* 1. Мотивация
  2. Цели и задачи
  3. Ограничения

## КРАТКО ОПИСАНИЕ на технологиите

### 1.4.1 Python

Избраният език за реализация на голяма част от тази дипломна работа е Python. Езикът позволява лесна работа с голямо количество библиотеки и структури от данни. Това помага с бърза реализация на сложни системи, използвайки само няколко реда програмен код.

Езикът за програмиране има употреба при статистически проучвания, заради лесната работа със структури от данни. Също е предпочитан за използване при работа с изкуствен интелект, заради простият си синтаксис. Това прави програми, написани на този език, лесни за четене и поддръжка. Езикът предоставя гъвгави инструменти за работа, голямо количество от библиотеки и голямо общество от разработчици, които го поддържат.

1.4.2 C# - .NET MAUI

Езикът C# е използван за реализация на мобилно приложение, което да позволява достъп до данните от системата за паркиране. Използвана е структурата за програмиране Multi-platform App UI (MAUI), която от своя страна стъпва върху структурата .NET.

Това е популярна структура, която позволява разработка на приложения, които могат да работят върху различни операционни системи, използвайки един и същи код. Структурата позволява дадено приложение да работи върху Android, iOS, Windows и macOS, без да са необходими промени в кода. Тази структура е наследникът на Xamarin.Forms. Тя разширява функционалността на старата структура, като добавя увеличена производителност и по-лесни за разширяване приложения.

Структурата MAUI работи като унифицира интерфейсите за програмиране на приложения на Android, iOS, Windows и macOS и предоставя собствени интерфейси, които сами определят коя платформа трябва да използват. [1]

1.4.3 PostgreSQL

PostgreSQL е система за управление на релационни бази данни с отворен код. Системата използва и разширява езикът Structured Query Language (SQL) и предоставя функционалност за сигурно съхранение на данни. Системата за управление може да работи под всички често използвани операционни системи, предоставя голямо количество инструменти за улеснено управление или разширяване на функционалност и има широк кръг от потребители, което прави решаването на проблеми лесно. [2]

1.4.4 SQLAlchemy

SQLAlchemy е библиотека за Python, която позволява лесна работа с голямо количество системи за управление на релационни бази данни. Библиотеката предоставя унифициран начин за достъпване на данните в базите данни, което ни позволява лесно да сменим типа на релационната база данни, без да се налага да се променя кодът, който използва базата данни. Това прави тази библиотека много ценна при създаване на каквото и да е приложение, което работи с база данни.

За да се създаде връзка с база данни, е необходимо да има известни адрес на сървъра на базата, порт на сървъра на базата, потребителско име, парола и име на базата данни. След това библиотеката сама се грижи за тази връзка и ние просто трябва да създаваме сесии, с които да работим.

1.4.5 FastAPI

FastAPI представлява библиотека на Python, която позволява изграждане на интерфейси за достъп до сървър. Библиотеката се работи директно с езика Python, което значи, че потребителят просто трябва да посочи парче код, което иска да използва като интерфейс за достъп, и библиотеката FastAPI сама се грижи за изграждането на интерфейсa.

След като се опишат входните точки на интерфейса, FastAPI може да генерира стандартизиран документ, който описва всички интерфейси и данни, които приема приложението, познат като OpenAPI файл. Този файл се използва от клиентски приложения за да се генерира код, който да позволи връзка с нашето приложение.

1.4.6 Uvicorn

Uvicorn e уеб сървър за Python. Представлява имплементация на Asynchronous Server Gateway Interface (ASGI) спецификацията. Тази спецификация описва интерфейси за изграждане на асинхронни и синхронни сървъри на Python.

Uvicorn е напълно съвместим с OpenAPI спецификацията и следователно е съвместим с FastAPI. Така в едно приложение можем да използваме FastAPI за да създадем OpenAPI съвместими входни точки за сървърът ни и след това да пуснем Uvicorn, който да обслужва входните точки.

1.4.7 Разпознаване на обекти

Разпознаване на обекти или Object Detection представлява технология, при която се тренира невронна мрежа, която да разпознава дадени обекти. Тази технология е техника за ‚компютърно зрение‘, което буквано представлява способността на дадена машина да ‚вижда‘. Ще навлезем по-дълбоко в невронни мрежи и разпознаване по-късно в тази глава.

1.4.8 YOLOv8

YOLOv8 или You Only Look Once представлява модел за разпознаване на обекти на Ultralytics, който е бърз, точен и лесен за употреба. Моделът може да класифицира обекти, да разпознава обекти, да сегментира обекти, да проследява обекти и да определя пози. Моделът има няколко различни размери, като по-големите модели са по-точни, но и изискват повече ресурси за да работят. Най-лекият модел е nano, последван от small, последван от medium, след това large и най-големият модел x. [3]

1.4.9 OCR

OCR или Оптично Разпознаване на Символи представлява технология, при която снимка на текст, било той ръчно написан или принтиран, се подава на конвертор, който превръща снимката в текст, разбираем от машина. Тази технология позволява автоматично въвеждане на данни от документи, което елиминира ръчното въвеждане и така увеличава скоростта на обработка на данни. Проблемът е, че снимката трябва да бъде ясна, за да може алгоритъмът да прочете данните коректно.

1.5 Разпознаване на обекти и машинно самообучение

### 1.1.2 Мотивация

От изброените методи за измерване на разстояние липсва такъв който отговаря на всички изисквания:

* Евтин
* Работи на разстояние от сантиметри до няколко метра
* Не е нужно твърде висока точност
* Безжичен
* Лесен за интегриране във вече съществуващи системи
* Не разчита на допълнителни сензори (акселерометър и жироскоп)

Разработената система покрива тези изсиквания чрез използването на ESP32 чипове. Те са широко популярни във средите на вградените микроконтролерни системи и интернет на нещата продукти. Тези микроконтролери са евтини и по-голямата част от тях имат вградена антена и позволяват работа с други безжични устройства на базата на   
Wi-Fi и Bluetooth. Система за измерване на разстоянията базирана на статистически анализ може да покрива различни разстояния в зависимост от силата на антената, нужната точност и времето, което сме склонни да отделим, за едно измерване. По този начин ние можем да създадем конфигуриуема система, която покрива нуждите на всеки потребител.

### 1.1.3 Потенциални препядствия

Използването на RSSI за измерване на разстояние създава трудности при измерване на по-големи разстояния. С увеличаването на разстоянието, силата на сигнала пада и са необходими повече измервания за по-висока точност. В даден момент е възможно да бъде загубена Wi-Fi връзката, което е възможно да доведе до невалидни данни от измерването.

Също проблем е че отчитането на силата на сигнала е нелинейна – разликата между 99dBm - 100dBm и 100dBm и 101dBm не е същата. Това въвежда допълнително ниво на сложност при конвертирането на данните, което може да затрудни работата на системата и забави изчисленията.

Друго препядствие при измерване на разстояние чрез RSSI е наличието на обекти между комуникиращите устройства. Всеки обект, поставен на пътя на сигнала, изкривява резултатът от измерването. При разработката на тази дипломна работа, това е взето предвид и се извършват многократни измервания за да се повиши точността, като се вземе резултатът с най-голяма вероятност за вярност. Този начин на работа ни позволява да имаме по-висока точност, заради повечето обменени данни и съответно повече направени измервания на силата на сигнала. Това може да ни помогне да избегнем препядствието, представено от обекти, пречещи на Wi-Fi сигнала, когато те не са прекалено големи или плътни. Когато имаме по-плътен или голям обект, не можем да получим точно измерване, тъй като силата на сигнала за всички обменени данни ще бъде силно повлияна от обекта, който пречи. Това води до изкривявания в измерването, които могат да варират силно, спрямо очакваните от нас резултати. Това създава ограничение върху тази технология, което изисква комуникиращите си устройства да бъдат поставени така, че да се минимизира потенциалът от интерференция от чужди обекти, които могат да попречат на силата на сигнала.

Въпреки че ESP32 е много популярен чип, той не е с лимитирани възможности. Първата потенциално критична лимитация е в количеството системните ресурси. В зависимост от модела тази серия микроконтролери те могат да имат между 500KB и 8MB рам, едно или две ядра работещи до 240MHz. Друга лимитация на тези микроконтролери идва с библиотеките за Wi-Fi и решенията за показване на RSSI стойностите и в частност това че те са целочислени числа, което изкривява реалните стойности на измерването.

## 1.2 Цели и задачи

**Целта на дипломната е работа е да се разработи система за измерване на разстояние която запълва нишата описана в 1.1.2. Тази система трябва да е евтина, лесна за интегриране и използване, да има лесен за употреба потребителски интерфейс, който да не претрупва потребителят с ненужни данни и да позволява лесна настройка на мрежата, използвана за измерване. Потребителският интерфейс също трябва да позволява лесна настройка на конфигурацията за измерване, като например броят съобщения, който се изпраща или резолюция за точността на измерването. Конфигурацията в приложението не трябва да бъде зависима от броя на настройките или броя на микроконтролерите, участващи в измерването. Разработената система е необходимо да бъде надеждна, в смисъл на безотказна работа, тъй като макар да не се нуждае от висока точност, тя трябва да не спира да докладва информация за разстояние, за да не се загуби информация от участващите мрежови възли.**

Самият системен софтуер трябва да притежава няколко критични характеристики, за да осигури оптимална производителност и поддръжка. Първо, мащабируемостта е от съществено значение, тъй като софтуерът трябва да може ефективно да обработва нарастващ брой потребители, данни и транзакции без значителен спад в производителността. За да се постигне това, архитектурата трябва да бъде проектирана така, че да поддържа лесно увеличаване на капацитета и функционалността. Освен това, софтуерът трябва да бъде модулен. Тази модулност гарантира, че отделните компоненти могат да бъдат разработвани, тествани и поддържани независимо, което подобрява надеждността и гъвкавостта на системата. По този начин се опростява откриването на проблеми и се улеснява актуализирането и подобрението на софтуера. Друго от съществено значение е обширната документация. Тя е важна както за разработчиците, така и за крайните потребители. Добре документираният софтуер улеснява въвеждането на нови разработчици, подпомага откриването на проблеми и поддръжката и осигурява, че потребителите могат ефективно да използват функционалностите на системата. Статичният анализ на кода е жизненоважен компонент на процеса на разработка. Системата не трябва да въвежда проблеми при статичния анализ на кода като излизане извън границите, дереференции на нулеви указатели, невалидни преобразувания на типове или течове на памет. Тези проблеми могат да доведат до уязвимости, сривове на системата и деградирана производителност, което е неприемливо за надежден софтуер. Комуникацията в системата трябва да бъде надеждна, осигурявайки точна и бърза доставка на съобщенията. Системата трябва да поддържа множество активни системи едновременно. Всяка система трябва да може да обработва комуникация с множество потребители едновременно, осигурявайки целостта и консистентността на данните във всички потребителски взаимодействия.

## 1.3 ОБХВАТ И ОГРАНИЧЕНИЯ

Основният фокус ще бъде върху вътрешни среди като офис сгради, домове и търговски обекти. Работата на системата в открити или силно динамични среди, като открити полета или движещи се превозни средства, няма да бъде обхваната подробно. Системата трябва да работи при липса на препятствия между възлите, които биха довели до неточни изчисления. Устройствата, които ще се използват, ще бъдат разработки от серията ESP32 на Espressif. Въпреки стремежа към висока точност, системата признава потенциалните неточности поради ефекти на многократен път, затихване на сигнала и шум. Количествен анализ на тези грешки и техниките за тяхното смекчаване ще бъдат част от оценката, но няма да бъдат основен фокус.

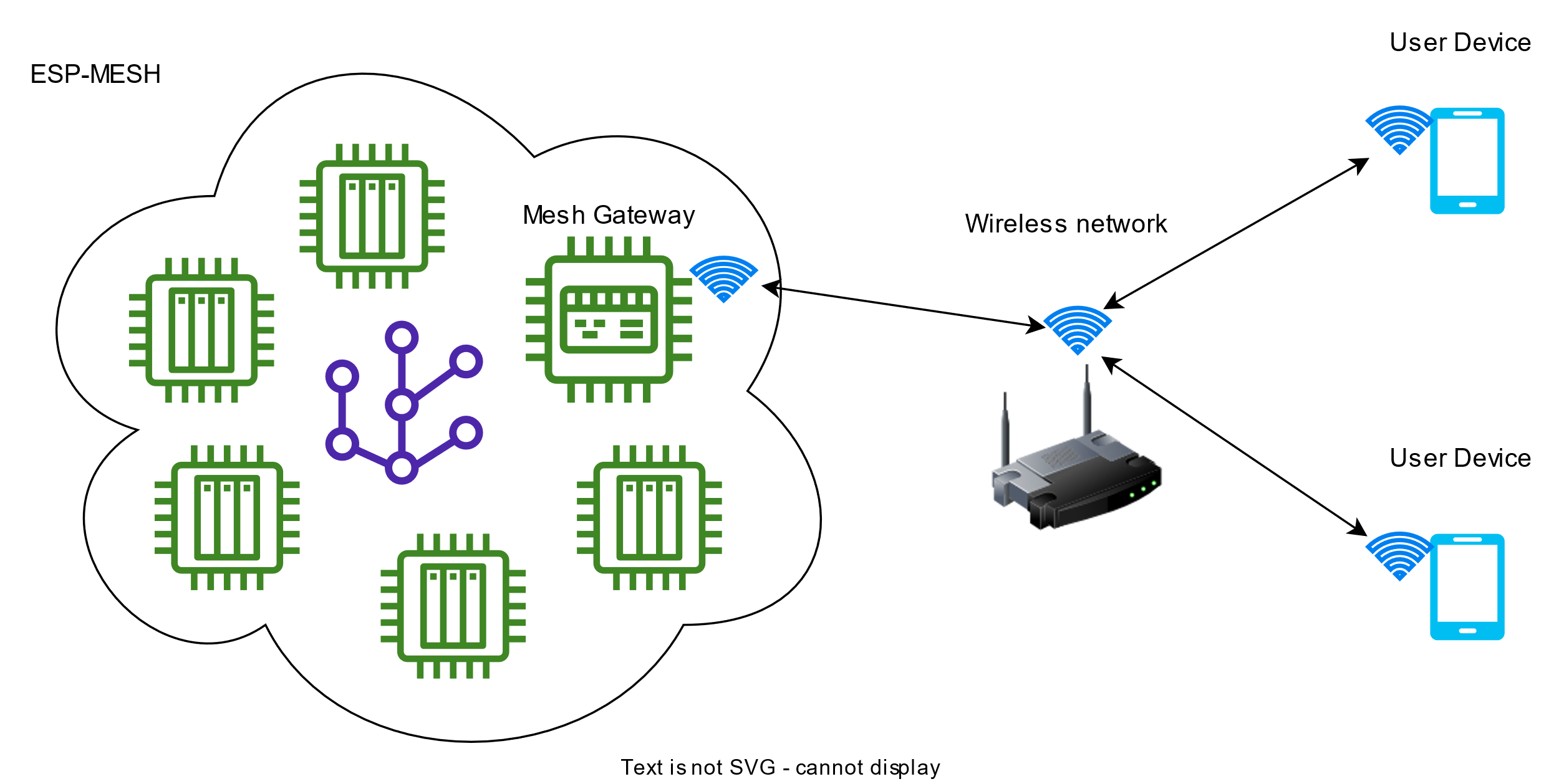
Системата ще използва протокола ESP-NOW на Espressif за изчисляване на разстоянието. Системата ще използва протокола ESP-MESH на Espressif за създаване на мрежа от възли. Системата ще се свърже с Wi-Fi и ще отвори порт за комуникация с потребителите. Ако не бъде намерена подходяща мрежа, системата ще създаде своя собствена мреж. Потребителското приложение трябва да бъде функционално както за Windows, така и за Android, така че ще бъде написано на .Net Maui. Системата може да бъде ограничена по отношение на точността и обработката на грешки. Възможно е версията на ESP32 с 500KB RAM да няма достатъчно памет за задачата. Ако е така, трябва да бъде заменена с версия с повече RAM.

# Функционално описание.

## 2.1 Преглед на системата

### 2.1.1 Описание на системата

Систечата за измерване на разстояние е съставена от 2 основни, независими, елемента. Това са потребителстото приложение и мрежата за измерване, които си говорят по TCP канал. Мрежата за измерване е съставена от неопределен брой устройства, които си общуват по един от два начина – чрез ESP-MESH за системни съобщения и през ESP-NOW за съобщения, съврани с измерването на разстоянието. Едно от Esp32 устройствата се повишава автоматично в корен на мрежата, който се грижи за предаване на комуникацията между мрежата и клиентските устройства. Този корен изпълнява функциите едновременно на шлюз (Gateway) и на нормален възел в мрежата. Компонентът от кода, който се занимава с нормалното функциониране на устройството като възел, не знае и не се интересува кое устройство е коренът – това на което се изпълнява кодът на сегашното устройство или някое друго устройстово в мрежата. На Фигура 1 е показан опростен поглед на системата.



Фигура 1

На фигурата е показана типична постановка на системата за измервания на разстояния, където има предварително установена мрежа от шест устройства. Едно от устройствата изпълнява ролята на шлюз, който е свързан към точка за достъп. Към същата точка за достъп са свързани две потребителски устройства, които могат да си говорят с мрежата.

### **2**.1.2 Работен процес

Работният процес на системата започва при включване на устройствата, които съсдават мрежата за измерване на разстояния. Първата стъпка е инициализилането на устроствата и започването на договарянето за създаването на мрежата. Създаването на мрежата е вградена част от протокола ESP-MESH и процедурите по създавене и подръжка са подробно описани в сайта на производителя. Тъй като, този протокол се нуждае от това да се върже към точка за достъп (access point), ако устройствата не успеят да се свържат към такава, то те започват работа като такава, което улеснява работата със систмата в случаите където няма точки за достъп. Втората стъпка е вписването на клиентското устройство в същата точка като мрежта за измерване и включването на клиентското приложение. След като клиентът установи връзка с мрежата за измерване, той може да изпраща съобщения към мрежата. Всички съобщения между мрежата и клиента минават през шлюза, като той има грижата да разпределя и препраща съобщенията и командите до всяко заинтересовано устройство, било то потребителско или участник в мрежата за измервания.

## 2.2 Key Components ot espto and Their Functions

## 

## 2.3 Communication Protocol

* Explain the communication protocol used for sending and receiving Wi-Fi messages. This might include details on the frequency of messages, handling of data packets, and any error correction mechanisms.

## 2.4 Data Collection and Processing

* **Data Collection**: Describe how data is collected from the Wi-Fi messages, including any preprocessing steps.
* **Statistical Analysis**: Explain the statistical methods used to analyze the collected data and calculate the most likely distance.
* **Accuracy and Calibration**: Discuss any techniques used to improve accuracy, such as calibration procedures or filtering algorithms.

## 2.5 User Interface

* Describe the design and functionality of the user interface. How does it display the distance measurements, and what options does it provide to the user for interacting with the system?

## 2.6 Use Cases

* Provide a brief description of the main use cases that will be detailed later in Chapter 4. This serves as a preview for the reader.

# Програмна реализация.

## 3.1 комуникационна архитектура

### 3.1.1 Комуникационна архитектура във възловата мрежа

### 3.1.2 Комуникационна архитектура между потребителско устройство и възловата мрежа

### 3.1.3 Комуникационна архитектура между възли за изчисляване на разстояние

## 3.2 Системна архитектура на потребителското приложение

## 3.3 Системна архитектура на възел

# Ръководство за използване и примери за употреба.

## 4.1 Примерна постановка при наличие на Wi-Fi мрежа

## 4.2 Примерна постановка без Wi-Fi мрежа

## 4.3 Настройване на системата

# Заключение

# Съкращения

8

MAUI - Multi-platform App UI 6

SQL - Structured Query Language 6

МПС - Моторни Превозни Средства 5

# Източници

|  |  |
| --- | --- |
| [1] | „What is MAUI,“ Microsoft, [Онлайн]. Available: https://learn.microsoft.com/en-us/dotnet/maui/what-is-maui?view=net-maui-8.0. |