Частна професионална гимназия по дигитални науки „Софтуни Светлина“, гр. София

ДИПЛОМЕН ПРОЕКТ

на …….. ученик/ученичка от ……. клас професия- код: „Приложен програмист“ - 481030 специалност- код: „Приложно програмиране“ - 4810301

Тема: RC количка

Изисквания за разработката на дипломния проект:  
Да се създаде приложение с помощта на една от следните технологии C#, PHP, Python или JavaScript. Системата трябва да се разработи като трислойно приложение, като разполага със слой за данни , слой за услуги , слой за потребителския интерфейс. За приложението трябва да се разработят компонентни тестове.

Ръководител-консултант: …………..

Сесия: ……………..

Дата:.........................

[1. Увод 3](#_Toc128427826)

[2. Изложение 4](#_Toc128427827)

[2.1. Цел на дипломния проект 4](#_Toc128427828)

[2.2. Използвани технологии 5](#_Toc128427829)

[2.2.1. Уеб приложение 5](#_Toc128427830)

[2.2.2. Софтуерни технологии 5](#_Toc128427831)

[2.2.2.1. Arduino IDE 5](#_Toc128427832)

[1.1.1.1. Arduino IDE Language (C++) 7](#_Toc128427833)

[1.1.1.2. Допълнителни Библиотеки 8](#_Toc128427834)

[9](#_Toc128427835)

[1.1.1.3. Настройване на точката за достъп 9](#_Toc128427836)

[1.1.1.4. Методи 10](#_Toc128427837)

[1.1.1.5. HTML 13](#_Toc128427838)

[1.1.1.6. CSS 14](#_Toc128427839)

[1.1.1.7. HTTP 15](#_Toc128427840)

[1.1.1.8. HTPP заявки 16](#_Toc128427841)

[1.1.1.9. REST 17](#_Toc128427842)

[1.1.2. Хардуерни технологии 18](#_Toc128427843)

[1.1.2.1. Node MCU ESP8266 18](#_Toc128427844)

[1.1.2.2. Широчинно-импулсната модулация – PWM 21](#_Toc128427845)

[1.1.2.3. H – модул драйвер за контрол на мотори 21](#_Toc128427846)

[1.1.2.4. L298N драйвер 23](#_Toc128427847)

[1.1.2.5. 150 RPM BO Motor 24](#_Toc128427848)

[2.3. Задачи , които трябва да се решат за постигането на целта 25](#_Toc128427849)

[2.4. Употреба на приложението 25](#_Toc128427850)

[3. Заключение 26](#_Toc128427851)

[4. Източници 26](#_Toc128427852)

[https://docs.arduino.cc 26](#_Toc128427853)

# Увод

Проекта ,, RC Remote Car’’ е разработка в средата на ,,Интернет на нещата(IoT)“. Интернет на нещата , известен още като интернет на вещите или интернет на обектите, е концепция за компютърна мрежа от физически обекти (устройства, превозни средства, сгради и други предмети и вещи), притежаващи вградени електронни устройства за взаимодействие помежду си или с външната среда. Тази концепция разглежда организацията на такива мрежи като явление, способно да преустрои икономическите и обществени процеси така, че да изключи необходимостта от участие на човека в част от действията и операциите. Колите с дистанционно управление са били популярна играчка от поколения, осигурявайки безкрайни часове забавление както за деца, така и за възрастни. През последните години навлизането на WiFi технологията доведе до ново поколение автомобили с дистанционно управление, които могат да се управляват от смартфон или компютър, осигурявайки още по-голяма гъвкавост и контрол. Един такъв пример за това е колата с дистанционно управление базира на ESP8266. ESP8266 е евтин микроконтролер с Wi-Fi, който може да се използва за управление на различни електронни устройства, включително автомобили с дистанционно управление. Проектът включва изграждане на кола с дистанционно управление и интегриране на микроконтролер ESP8266, за да се даде възможност за безжично управление. Микроконтролера е програмиран да се създава собствена WiFi мрежа(уеб сървър) на default IP: 192.168.4.1 и да получава команди от отдалечено устройство, като смартфон или компютър. След това тези команди се преобразуват в команди за движение за двигателите на автомобила, позволявайки автомобилът да се управлява дистанционно. Като цяло колата с дистанционно управление ESP8266 WiFi е забавен и предизвикателен проект, който може да се хареса както на любители, студенти, така и на професионалисти. Той съчетава най-новата технология с класическия автомобилен дизайн с дистанционно управление, за да създаде една наистина уникална и многостранна система. Независимо дали се интересувате от електроника, програмиране или просто обичате да играете с коли с дистанционно управление, колата с дистанционно управление ESP8266 WiFi е проект, който със сигурност ще ви осигури часове забавление и образование.

# Изложение

## Цел на дипломния проект

Главната цел на проекта е да предостави въведение в различни технологии и концепции, включително микроконтролери, безжични мрежи и дистанционно управление. Изисква основно разбиране на програмирането и електрониката, което го прави чудесен проект за тези, които искат да научат повече за тези области. Като цяло, целта на проекта за кола с дистанционно управление ESP8266 WiFi е да предостави забавен и предизвикателен проект, който може да се хареса както на всички. Той съчетава най-новата технология с класическия автомобилен дизайн с дистанционно управление, за да създаде една наистина уникална и многостранна система.

## Използвани технологии

### Уеб приложение

Управлението на количката се случва от семпло, бързо и ефективно уеб приложение което се държи на уеб сървъра на микроконтролера.

Уеб приложение е програма, до която потребителите имат достъп чрез мрежа като Интернет. Поддържан от браузър език за програмиране (като JavaScript), комбиниран с изобразен от браузър език за маркиране (като HTML), също може да се използва за описание на софтуерно приложение, което се изобразява от типичните уеб браузъри.

Уеб приложенията са широко използвани поради широкото разпространение на уеб браузъри. Тъй като използват както уеб браузъри, така и уеб технологии за извършване на специфични операции през Интернет, тези технологии имат огромна целева аудитория.

### Софтуерни технологии

#### Arduino IDE

Интегрираната среда за разработка на Arduino (IDE) е софтуерно приложение, което се използва за създаване и качване на код към платките на микроконтролери. Това е удобна за потребителя платформа, която предоставя прост и интуитивен интерфейс за програмиране и отстраняване на грешки на платки. Графични интерфейс е базиран на Eclipse Theia.

Arduino IDE е софтуер с отворен код, който може да бъде изтеглен безплатно от уебсайта на Arduino. Той е съвместим с множество операционни системи, включително Windows, macOS и Linux, и поддържа широка гама от платки, включително популярните Arduino, ЕSP и т.н.

Основните характеристики на Arduino IDE включват:

* Сериен монитор: Серийният монитор е вграден инструмент в Arduino IDE, който ви позволява да комуникирате с вашата Arduino платка през серийния порт. Той може да показва данни в реално време от вашата платка Arduino и можете също да изпращате данни обратно към платката с помощта на серийния монитор.
* Мениджър на борда: Мениджърът на борда е инструмент в Arduino IDE, който ви позволява да добавяте и управлявате различни типове платки. Той включва предварително конфигурирани настройки за популярни платки, което улеснява настройването на вашата платка и започването на програмиране.

В обобщение, Arduino IDE е мощен инструмент, който опростява процеса на програмиране на различни платки. Неговият интуитивен интерфейс и вградени инструменти улесняват както начинаещите, така и напредналите потребители да създават и качват код на своите микроконтролери.



Фигура - Ардуино IDE

#### Arduino IDE Language (C++)

Езикът за програмиране Arduino е опростена версия на езика за програмиране C++. Той е специално проектиран да улесни начинаещите да започнат с програмирането на микроконтролери като тези, които се намират на платките Arduino.

Програмният език Arduino е подобен на C++ по много начини, но включва редица вградени функции и библиотеки, които улесняват взаимодействието с хардуера на платките. Например, има вградени функции за управление на цифрови и аналогови пинове, сензори за четене и комуникация с други устройства.

Ето някои от основните характеристики на езика за програмиране Arduino:

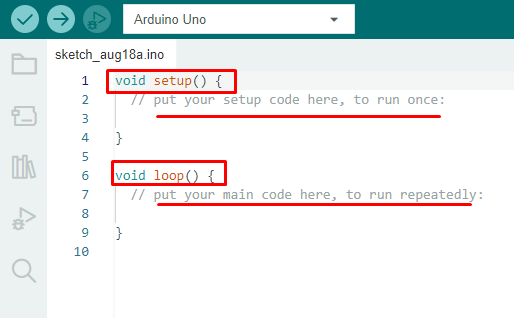
Setup() и Loop(): Всяка скица на Arduino има две основни функции: setup() и loop(). Функцията setup() се извиква веднъж, когато платката се включи или нулира, и се използва за инициализиране на променливи, задаване на режими на щифтове и конфигуриране на платката. Функцията loop() се извиква многократно, докато платката работи, и се използва за контролиране на основното поведение на платката.

Режими на щифта: В езика за програмиране Arduino можете да зададете режима на щифта на INPUT или OUTPUT. Това ви позволява да контролирате дали щифтът се използва за четене на данни от сензор или за изпращане на данни към друго устройство.

Цифрови и аналогови I/O: Езикът за програмиране Arduino включва вградени функции за управление на цифрови и аналогови пинове на платката. Това позволява да се контролират светодиоди, двигатели и други устройства, които изискват цифрови или аналогови сигнали.

Библиотеки: Езикът за програмиране Arduino включва редица вградени библиотеки, които могат да се използват за опростяване на общи задачи, като управление на LCD дисплеи, сензори за четене и комуникация с други устройства.

Като цяло езикът за програмиране Arduino е проектиран да бъде лесен за научаване и използване, дори и за хора без опит в програмирането. Той позволява да взаимодействате с хардуера на платките по прост и интуитивен начин, което го прави чудесен избор за създаване на прототипи и експериментиране с електронни проекти.



Фигура - Ардуино код

#### Допълнителни Библиотеки

Природата с отворен код на проекта Arduino улесни публикуването на много безплатни софтуерни библиотеки, които други разработчици използват, за да разширят своите проекти.

Библиотеките или header-ите са файлове, написани на C или C++ (.c, .cpp), които предоставят на проектите допълнителна функционалност (напр. възможност за управление на LED матрица или четене на енкодер и т.н.).

Библиотеките използвани в дипломната работа са ESP8266WiFi.h и ESP8266WebServer.h.

Класът WebServer, открит в библиотеката ESP8266WebServer.h, е семпъл уеб сървър, който знае как да обработва HTTP заявки като GET и POST и може да поддържа само един едновременен клиент, за да избегнем случай ако две устройства се вържат към нашата количка. Сървърът работи на порта по подразбиране (порт 80).

Wi-Fi библиотеката за ESP8266 е разработена на базата на ESP8266 SDK, като се използват конвенциите за именуване и цялостната функционална философия на Arduino WiFi библиотеката. Тази библиотека предоставя специфични за ESP8266 Wi-Fi процедури, които извикваме, за да се свържем с мрежата.

#### ESP8266WiFi library — ESP8266 Arduino Core 3.1.1-19-gbe02af05 documentation

Фигура - Точка за достъп

#### Настройване на точката за достъп

Фигура - Настройка на точка на достъп

A picture containing text

Description automatically generated

С тези няколко реда настройваме настройките на микроконтролера. Задаваме му локален IP адрес, gateway и subnet маска.

Локални IP адреси се използват само в рамките на дадена локална мрежа. Крайните потребители в 99% от случаите използват именно такива адреси, защото няма възможност да се даде IP адрес на всяко едно устройство в къщата.

Gateway или шлюз е рутер, прокси сървър или софтуер, който свързва различни телекомуникационни мрежи, използващи несъвместими протоколи (например локални и глобални).

Subnet mask или подмрежовата маска е 32-битов адрес, който разделя IP адреса на мрежови битове, които идентифицират мрежата, и хост битове, които идентифицират хост устройството, работещо в тази мрежа. Той капсулира набор от IP адреси, които една подмрежа може да използва, като подмрежата се отнася до по-малка мрежа в рамките на по-обширна мрежа. Главно се използват вътрешно в мрежата.

Text

Description automatically generated

Фигура - настройка на сървъра

Също така трябва да кажем на сървъра да „слуша“ за дадени заявки. Първо вкарваме микроконтролера в режим на точка за достъп и конфигурираме всички параметри на мрежата. След това сървъра е ще извика дадена функция при искано URI от потребителя. Например функцията server.on приема три параметъра (URI, вид на заявката, функция която да се извика при код 200).

URI на български означава унифициран идентификатор на ресурс. URI е символен стринг, който препраща към даден ресурс. Той позволява да се идентифицира някакъв определен ресурс: документ, изображение, файл, служба, кутия на електронна поща и т.н. Преди всичко, става въпрос за мрежови ресурси от интернет и World Wide Web.

#### Методи

В контекста на програмирането, функция (метод) се нарича именувана група от инструкции, които изпълняват дадена операция (функционалност). Тази група от инструкции е логически отделена и именувана, така че изпълнението на инструкциите в групата може да бъде стартирано чрез извикване на нейното име в хода на изпълнението на програмата.

A picture containing text

Description automatically generated

Фигура - Void функция

Имаме няколко void метода които не връщат стойности, те отговарят за изпращането на заявките, зареждането на уеб приложението и за променливата car\_mode която отговаря за режима на колата дали ще върви напред, назад, наляво, надясно и т.н. .

Text

Description automatically generated

От друга страна имаме метод като SendHTML, който изпраща html и CSS под формата на стринг към даденото URI при HTTP отговор 200.

A picture containing calendar

Description automatically generated

И вече имаме още функция за car\_mode променливата, която според определя режим на колата, пуска моторите в дадените посоки.

Text

Description automatically generated with medium confidence

Имаме и Setup метод, който служи за настройване не само на уеб сървъра но и на самите мотори на количката. Това го постигаме с функцията pinMode като подаваме номера на пина който е дефинирам глобално и после даваме на какъв режим ще работи, т. е. ще дава или ще получава информация.

#### HTML

HTML (HyperText Markup Language) е най-основният градивен елемент на мрежата. Той определя значението и структурата на уеб съдържанието. Други технологии освен HTML обикновено се използват за описване на външния вид/презентацията (CSS) или функционалността/поведението на уеб страницата (JavaScript).  
„Хипертекст“ се отнася до връзки, които свързват уеб страници една с друга, или в рамките на един уеб-сайт, или между уеб-сайтове. Връзките са основен аспект на мрежата. Като качвате съдържание в Интернет и го свързвате със страници, създадени от други хора, вие ставате активен участник в World Wide Web.  
HTML използва "маркиране" за анотиране на текст, изображения и друго съдържание за показване в уеб браузър.

Описанието на документа става чрез специални елементи, наречени HTML елементи или техните маркери, които се състоят от тагове и съответстващите етикети (HTML tags) и ъглови скоби (като например елемента <html>). HTML елементите са основната градивна единица на кода, който изграждат уеб страниците. Чрез тях се форматира, графично оформя текста и неговите отделните части в рамките на една уеб страница, като например заглавия, цитати, текстови раздели, хипертекстови препратки и т.н. Най-често HTML елементите са групирани по двойки <h1> и </h1>.

Фигура - HTML

#### CSS

CSS (Cascading Style Sheets) е език за описание на стилове (език за стилови файлове, style sheet language) – използва се основно за описание на онлайн представянето на уеббазиран документ, който написан на език за маркиране. Най-често се използва допълнително към чистия HTML, но се прилага и върху XML уебстраници и документи. Спецификацията на CSS официално се поддържа от W3C.

CSS още в началото на развитието на www започва да се добавя към стандартния HTML с цел да бъдат разделени съдържанието и структурата на уеб страниците отделно от тяхното визуално представяне.

CSS може да се пише не само в отделни файлове но и в style тагове, като в html стринга на програмата.

|  |
| --- |
|  |
|  |



Фигура - CSS

#### HTTP

HTTP протоколът е измислен със създаването на World Wide Web (световната информационна мрежа или уеб). Този протокол определя правилата, по които се извършва обмяната на информация между уеб сървър и уеб клиент (най-често уеб браузър).

В HTTP протокола се използват понятия като клиент (обикновено това са Web браузърите (или web навигаторите) – т.е. самите приложения, a не физическите хостове в мрежата) и сървър (това са уеб сървърите – т.е. самите приложения, а не хостовете). HTTP Header Fields (HTTP хедъри) са полета с информация, намиращи се в заглавната част на HTTP заявката и HTTP отговора, които браузърът и уеб сървърът си разменят.

HTTP хедърите съдържат указания и параметри, от които зависи HTTP обмена на информация между клиент и сървър.



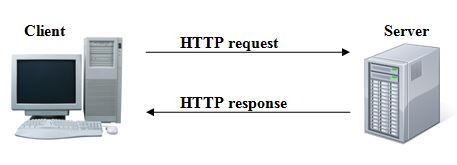
Фигура - HTTP

#### HTPP заявки

HTTP определя 8 различни клиентски метода за заявки:

* HEAD – иска изпращане на заглавията отговарящи на посочения с URL ресурс. Отговорът на сървъра е идентичен с този на GET, но е с липсващо тяло. По този начин може да се вземе само метаинформацията, свързана с ресурса, спестявайки трансфера на целия файл.
* GET – с него клиентът прави заявка за ресурс, зададен чрез URL. Могат да се изпращат и ограничено количество данни, закодирани директно в самия URL (отделени чрез въпросителен знак).
* POST – позволява клиентът да изпрати данни на сървъра. Тази заявка обикновено се генерира при изпращането на уеб формуляр, а данните могат да бъдат: текст, написан от потребителя във формуляра; файл на клиентския компютър и др.
* PUT – качва файл, който в бъдеще ще отговаря на посочения URL.
* DELETE – изтрива посочения ресурс.
* TRACE – сървърът връща получената заявка със статус OK. Позволява да се провери в какъв вид пристига заявката при сървъра и дали (и как) е била модифицирана по трасето от междинни прокси сървъри.
* OPTIONS – сървърът трябва да отговори с поддържаните от него клиентски методи, съответстващи на зададения URL, или с поддържаните от сървъра методи като цяло, ако е зададено \* вместо URL.
* CONNECT – използва се при комуникация през прокси.

Фигура - HTTP заявка



#### REST

Архитектурата REST е разработена успоредно с HTTP 1.1. Въпреки това, REST е обща архитектура, която може да бъде реализирана в други среди, а не само под HTTP.  REST е стил софтуерна архитектура за реализация на уеб услуги. Основната идея е да се определи системен ресурс, който се променя в резултат на взаимодействието между доставчика на услуги и потребителя. Архитектурният модел REST включва взаимодействията между сървър и клиент, осъществени по време на трансфера на данни. Концепцията беше въведена за пръв път от Рой Филдинг през 2000 г. като част от неговата докторска дисертация. Филдинг е един от основните автори на HTTP протокола, под който се изпълняват REST имплементациите в повечето случаи.



Фигура - REST

### Хардуерни технологии

#### Node MCU ESP8266

NodeMCU е изграден с ESP8266 базиран WiFi модул. Има 10 GPIO порта, 4MB Flash, два бутона, micro USB конектор и вградена PCB антена.

Платката е с размери 48mm x 25mm, с 4 монтажни отвора и с разстояние между изводите 2,54mm - съвместима с Breadboard. Захранва се с напрежение от USB 5V или от външен захранващ източник 5V - 10V, свързан на пина Vin – батерия, DC адаптер. Работното напрежение на GPIO портовете е 3,3V, при връзка с устройства и микроконтролери работещи на 5V, е необходимо да се използва конвертор на логически нива.

NodeMCU ESP8266 е популярна микроконтролерна платка, базирана на Wi-Fi модула ESP8266. Това е многофункционална и лесна за използване платка за разработка, която е предназначена да опрости процеса на изграждане на проекти за интернет на нещата (IoT). Със своята ниска цена, малък форм-фактор и вградена Wi-Fi свързаност, NodeMCU ESP8266 се превърна в любим сред любителите, производителите и професионалистите.

NodeMCU ESP8266 е платка с отворен код, която е базирана на скриптовия език Lua, което го прави лесен за програмиране и използване. Платката разполага с мощен Wi-Fi чип ESP8266, който може да се свързва с интернет и да комуникира с други устройства. Той има малък форм фактор, с размери само 49 mm x 24 mm, което го прави лесен за интегриране в малки проекти.

Една от ключовите характеристики на NodeMCU ESP8266 е неговата вградена Wi-Fi свързаност. Платката може да се използва за свързване към Wi-Fi мрежи и комуникация с други устройства в мрежата, включително сървъри, рутери и други IoT устройства. Това го прави идеален за IoT проекти, тъй като ви позволява да създавате проекти, които могат да бъдат контролирани и наблюдавани дистанционно през Интернет.

В допълнение към Wi-Fi свързаността, NodeMCU ESP8266 разполага с набор от входно/изходни (I/O) щифтове, които могат да се използват за свързване към сензори, задвижващи механизми и други компоненти. Платката има общо 9 I/O пина с общо предназначение (GPIO), които могат да се използват за цифров вход и изход, както и 1 аналогов входен пин.

NodeMCU ESP8266 също е оборудван с USB интерфейс, който може да се използва за захранване на платката и качване на код. Това улеснява програмирането и тестването на проекти, тъй като можете просто да свържете платката към вашия компютър и да използвате Arduino IDE или други инструменти за програмиране, за да качите код.

NodeMCU ESP8266 има широк спектър от приложения и може да се използва в различни проекти. Някои от най-често срещаните приложения включват домашна автоматизация, мониторинг на околната среда и дистанционно управление на устройства. Също така често се използва в проекти „Направи си сам“, като роботи, дронове и други електронни устройства.

В заключение, NodeMCU ESP8266 е универсална и лесна за използване микроконтролерна платка, която е идеална за IoT проекти и DIY електроника. Със своята ниска цена, вградена Wi-Fi свързаност и мощен Wi-Fi чип ESP8266, той се превърна в любим сред производителите и професионалистите. Независимо дали изграждате система за домашна автоматизация, наблюдавате условията на околната среда или създавате свой собствен проект „Направи си сам“, NodeMCU ESP8266 е отличен избор за следващия ви проект.



Фигура - Node MCU ESP8266

#### Широчинно-импулсната модулация – PWM

Широчинно-импулсната модулация (PWM) или импулсно-продължителната модулация (PDM) е метод за намаляване на средната мощност, доставяна от електрически сигнал, чрез ефективното му нарязване на отделни части. Средната стойност на напрежението (и тока), подадена към товара, се контролира чрез бързо включване и изключване на превключвателя между захранване и натоварване. Колкото по-дълго превключвателят е включен в сравнение с периодите на изключване, толкова по-висока е общата мощност, подадена към товара. ШИМ е особено подходящ за работа с инерционни товари като двигатели, които не се влияят толкова лесно от това дискретно превключване, тъй като тяхната инерция ги кара да реагират бавно. Честотата на превключване на ШИМ трябва да бъде достатъчно висока, за да не повлияе на товара, което означава, че резултантната форма на вълната, възприемана от товара, трябва да бъде възможно най-плавна.

A picture containing shape

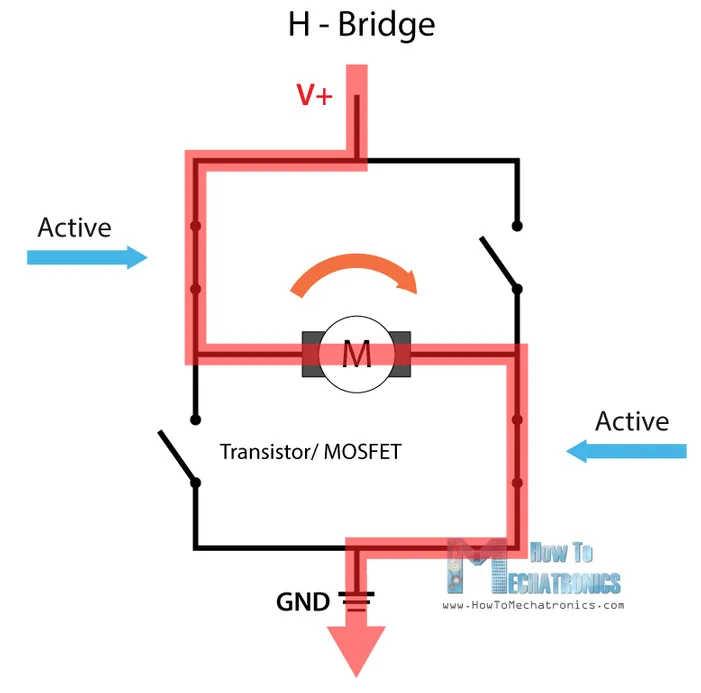
Description automatically generated

Фигура - ШИМ

#### H – модул драйвер за контрол на мотори

От друга страна, за да контролираме посоката на въртене, просто трябва да обърнем посоката на тока през двигателя и най-често срещаният метод за това е чрез използване на H-мост. Веригата H-Bridge съдържа четири превключващи елемента, транзистори или MOSFET, като моторът в центъра образува H-подобна конфигурация. Чрез активиране на два конкретни превключвателя едновременно можем да променим посоката на текущия поток, като по този начин променим посоката на въртене на двигателя.

Така че в зависимост от размера на двигателя, можем просто да свържем изхода на ESP PWM към основата на транзистора или портата на MOSFET и да контролираме скоростта на двигателя, като контролираме изхода PWM. Сигналът с ниска мощност на ESP PWM включва и изключва портата на MOSFET, през който се задвижва моторът с висока мощност.



Фигура - H Bridge

#### L298N драйвер

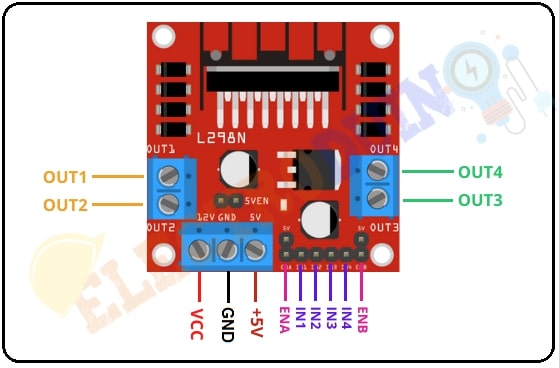
L298N е двигателен драйвер с двоен H-Bridge, който позволява управление на скоростта и посоката на два DC мотора едновременно. Модулът може да управлява постояннотокови двигатели с напрежение между 5 и 35 волта, с пиков ток до 2 ампера.

Модулът има два винтови клемореда за двата мотора и друг винтов клеморед за заземителния щифт, VCC за мотора и 5 волтов щифт, който може да бъде вход или изход.

Това зависи от напрежението, използвано при двигателите VCC. Модулът има вграден 5V регулатор, който се активира или деактивира с помощта на джъмпер. Ако захранващото напрежение на двигателя е до 12V, можем да активираме 5V регулатора и 5V щифтът може да се използва като изход, например за захранване на нашата Arduino платка. Но ако напрежението на двигателя е по-голямо от 12V, трябва да изключим джъмпера, защото тези напрежения ще причинят повреда на бордовия 5V регулатор. В този случай щифтът 5V ще се използва като вход, тъй като трябва да го свържем към 5V захранване, за да работи правилно IC(интегрална схема -  електронна схема с миниатюрни размери, състояща се от полупроводникови устройства и пасивни компоненти) .

Следват входовете за логическо управление. Щифтовете за включване на моторите се използват за активиране и контролиране на скоростта на двигателя. Ако на този пин има джъмпер, моторът ще бъде активиран и ще работи на максимална скорост, а ако махнем джъмпера, можем да свържем ШИМ вход към този пин и по този начин да контролираме скоростта на мотора. Ако свържем този щифт към заземяване, моторът ще

бъде деактивиран.



Фигура - L298 драйвер

#### 150 RPM BO Motor

Bo мотор (захранван от батерии) лек двигател с постоянен ток, който осигурява добър въртящ момент и обороти при по-ниски напрежения. Тези двигатели обикновено работят с напрежение между 3V и 12V, въпреки че точният обхват на напрежението може да варира в зависимост от конкретния модел на двигателя и производителя.

Двигателите BO могат да се управляват с помощта на моторен драйвер, който осигурява необходимия ток и напрежение към двигателя. PWM (широчинно-импулсна модулация) сигнали могат да се използват за управление на скоростта на двигателя, докато посоката може да се контролира с помощта на H-мостов драйвер.



Фигура - BO двигател

## Задачи , които трябва да се решат за постигането на целта

* Проектиране на дизайна на количката
* Сглобяване на количката
* Подготовка за писане на код (допълнителни библиотеки)
* Проектиране на структурата на уеб сървъра
* Тестване на локалната мрежа
* Проектиране на HTTP функционалности за изпращане на HTML към сървъра

## Употреба на приложението

Пускане на Количката - снимка

Връзване към точката за достъп – снимка

Отваряне на уеб приложението – снимка

# Заключение

В заключение, дипломния проект е забавен и вълнуващ проект, който съчетава силата на платката за микроконтролер на Nodemcu ESP8266 с тръпката от състезания с автомобили за дистанционно управление. Изграждайки този проект, вие сте натрупали ценен опит в електрониката, програмирането и механичния дизайн и сте научили как да интегрирате платката ESP8266 с други компоненти, за да създадете напълно функционален автомобил за дистанционно управление, който може да бъде контролиран с помощта на смартфон или таблет.

Този проект не само е чудесен начин да научите нови умения и техники, но е и практично приложение на IoT технологията, демонстрирайки как може да се използва за контрол на физическите устройства дистанционно по интернет. С автомобила за дистанционно управление ESP8266 вие създадохте проект, който е едновременно забавен и образователен и който има потенциал да вдъхнови бъдещи проекти и иновации.

Като цяло този проект е възнаграждаващ опит, който ви позволи да проучите възможностите на борда на микроконтролерите ESP8266 и света на IoT. С новите си знания и умения вече сте оборудвани да предприемете още по -амбициозни проекти и да допринасяте за нарастващата общност от създатели и новатори.

# Източници

<https://docs.arduino.cc>

<https://en.wikipedia.org/wiki/HTML>

<https://en.wikipedia.org/wiki/CSS>

<https://en.wikipedia.org/wiki/HTTP>

<https://components101.com/modules/l293n-motor-driver-module>