SŠPU
STŘEDNÍ ŠKOLA PRŮMYSLOVÁ
A UMĚLECKÁ, OPAVA

ZÁVĚREČNÁ STUDIJNÍ PRÁCE

dokumentace

Android aplikace pro ovládání auta řízeného ESP32

Pavel Fabián

Obor: 18-20-M/01 INFORMAČNÍ TECHNOLOGIE
se zaměřením na počítačové sítě a programování

Třída: IT4

Školní rok: 2024/2025

Poděkování

Rád bych poděkoval panu učiteli Godovskému za jeho cenné odborné rady při návrhu a zapojení elektronických součástek použitých v tomto projektu. Zvláštní poděkování mu patří také za zapůjčení potřebných komponent a pomoc s jejich úpravou, což významně usnadnilo celý proces vývoje. Jeho podpora a odborné vedení byly neocenitelné a přispěly k úspěšnému dokončení této práce.

Prohlašuji, že jsem závěrečnou práci vypracoval samostatně a uvedl veškeré použité informační zdroje.

Souhlasím, aby tato studijní práce byla použita k výukovým účelům na Střední průmyslové a umělecké škole v Opavě, Praskova 399/8.

V Opavě 09. 01. 2025

podpis autora práce

ABSTRAKT

Tato práce se zabývá vývojem Android aplikace pro dálkové ovládání modelu auta řízeného mikrokontrolerem ESP32-CAM. Cílem je vytvořit funkční systém pro bezdrátové řízení, přenos obrazu z kamery a zpracování uživatelských příkazů v reálném čase. Aplikace byla vyvinuta v Android Studio pomocí jazyka Java se zaměřením na přehledné uživatelské rozhraní a stabilní komunikaci. Mikrokontroler ESP32-CAM byl naprogramován v Arduino IDE pro přenos videa a zpracování příkazů přes Wi-Fi. Dokumentace popisuje klíčové technologie, použité komponenty a postupy vývoje. Důraz je kladen na efektivní bezdrátové spojení, přenos videa a rychlou odezvu systému. Závěrem jsou shrnuty výsledky testování, nalezené problémy a možnosti rozšíření. Projekt představuje funkční prototyp, který kombinuje moderní technologie s praktickým využitím v oblasti dálkového ovládání a IoT.

ABSTRACT

This work focuses on developing an Android application for remote control of a car model powered by the ESP32-CAM microcontroller. The goal is to create a functional system for wireless control, camera image transmission, and real-time user command processing. The application was developed in Android Studio using Java, emphasizing a user-friendly interface and stable communication. The ESP32-CAM microcontroller was programmed in Arduino IDE for video streaming and command processing via Wi-Fi. The documentation describes key technologies, components used, and development procedures, focusing on effiient wireless connection, video transmission, and system responsiveness.

Finally, testing results, identified issues, and expansion possibilities are summarized. The project demonstrates a functional prototype combining modern technologies with practical applications in remote control and IoT systems.

Obsah

ÚVOD **6**

1 VYTVÁŘENÍ MOBILNÍCH APLIKACÍ PRO OVLÁDÁNÍ IOT ZAŘÍZENÍ **7**

2 VYUŽITÉ TECHNOLOGIE **8**

2.1 HARDWARE 8

2.1.1 ESP32-CAM *8*

2.1.2 H-bridge L298N *8*

2.1.3 Motory *8*

2.2 SOFTWARE 9

2.2.1 Arduino IDE *9*

2.2.2 Android studio *9*

2.2.3 Inventor *9*

2.2.4 Prusa Slicer *9*

3 ZAPOJENÍ HARDWARU **10**

3.1 SCHÉMA A POPIS ZAPOJENÍ 10

3.1.1 ESP32-CAM *10*

3.1.2 H-Bridge *10*

3.1.3 Motory *11*

3.2 NAPÁJENÍ 11

4 APLIKACE **12**

4.1 ZÍSKÁVÁNÍ VIDEA 12

4.1.1 Video *12*

4.1.2 Problém využitého řešení *12*

4.2 KOMUNIKACE S ESP32 12

4.3 UŽIVATELSKÝ VSTUP 13

4.3.1 Zadání IP adresy *13*

4.3.2 Kontrola *13*

5 ŘÍZENÍ AUTA – KÓD ESP **15**

5.1 PŘIPOJENÍ K SÍTI 15

5.2 WEB SERVER PRO STREAMOVÁNÍ VIDEA 15

5.3 WEB SOCKET 16

5.4 OVLÁDÁNÍ MOTORŮ 17

6 MODEL A KONSTRUKCE DRŽÁKU ESP 18

6.1 MODELY 18

První díl 18

Druhý díl 18

Třetí díl 18

6.2 KONSTRUKCE 19

ZÁVĚR 20

SEZNAM POUŽITÝCH INFORMAČNÍCH ZDROJŮ 21

ÚVOD

Tato dokumentace se věnuje návrhu a realizaci mobilní aplikace pro ovládání autíčka řízeného pomocí ESP32, což zahrnuje přenos živého videa, ovládání směru pohybu, řízení LED diody a možnost správy základních nastavení. Zvolená problematika vychází ze zájmu o oblast internetu věcí (IoT) a propojení softwarového vývoje s hardwarovými systémy. Výběr tohoto tématu byl motivován osobní snahou naučit se programovací jazyk Java a osvojit si základy vývoje mobilních aplikací. Zároveň šlo o příležitost propojit tyto znalosti s hardwarovou částí projektu a prozkoumat způsoby komunikace mezi zařízeními, konkrétně pomocí WebSocket technologie pro rychlou výměnu dat mezi mobilní aplikací a ESP32.

Cílem projektu bylo vytvořit intuitivní a funkční řešení, které umožní uživateli vzdáleně ovládat zařízení ESP32 prostřednictvím mobilního telefonu. Aplikace byla navržena tak, aby poskytovala přehledný uživatelský zážitek, zahrnující živý přenos obrazu z kamery, ovládání směru pohybu pomocí směrových tlačítek, jednoduché zapínání a vypínání LED diody. Tyto funkce byly definovány s ohledem na požadavky a možnosti, které projekt nabízí, a zároveň byly zamýšleny jako základ pro případné budoucí rozšíření.

Dokumentace nabízí ucelený pohled na celý proces vývoje aplikace, od prvotního návrhu přes implementaci až po testování a optimalizaci. Text přibližuje nejen technické aspekty, ale také výzvy spojené s propojením softwaru a hardwaru a praktické zkušenosti získané během vývoje. Postupně se zaměřuje na výběr a využití konkrétních technologií, podrobný popis architektury a funkcí aplikace, a nakonec na vyhodnocení dosažených výsledků. Pro čtenáře, kteří se chtějí inspirovat nebo sami uvažují o podobném projektu, dokumentace přináší užitečné poznatky a možnosti dalšího využití nebo vylepšení.

1 VYTVÁŘENÍ MOBILNÍCH APLIKACÍ PRO OVLÁDÁNÍ IOT ZAŘÍZENÍ

Vývoj mobilních aplikací v oblasti IoT propojuje digitální svět s fyzickými zařízeními a nabízí široké možnosti pro jejich ovládání a monitorování. Tato práce se zaměřuje na vytvoření aplikace umožňující ovládání zařízení ESP32, které je známé svou podporou bezdrátové komunikace a nízkou spotřebou energie. Dosavadní poznatky v této oblasti ukazují, že ESP32 je díky své flexibilitě a dostupnosti jedním z nejpoužívanějších řešení v IoT projektech. Mobilní aplikace, vyvíjené pro platformu Android, umožňují vytvoření intuitivního uživatelského rozhraní a jednoduché propojení s hardwarem.

Klíčovým aspektem je volba protokolu pro komunikaci mezi aplikací a zařízením. WebSocket protokol byl zvolen pro svou schopnost poskytovat obousměrnou komunikaci s nízkou latencí, což je důležité například při ovládání motorů nebo LED diod a zvyšuje uživatelský komfort a efektivitu systému.

Cílem této práce je vytvoření funkčního systému, který zahrnuje mobilní aplikaci pro Android a hardwarovou platformu ESP32. Systém má zajistit spolehlivou a rychlou komunikaci, umožnit intuitivní ovládání IoT zařízení a poskytovat uživateli přístup k živému přenosu obrazu. Hypotéza práce předpokládá, že kombinace ESP32 a WebSocket protokolu je schopna zajistit požadovanou funkcionalitu v reálném čase, a že mobilní aplikace vytvořená v Android Studiu nabídne uživatelsky přívětivé prostředí.

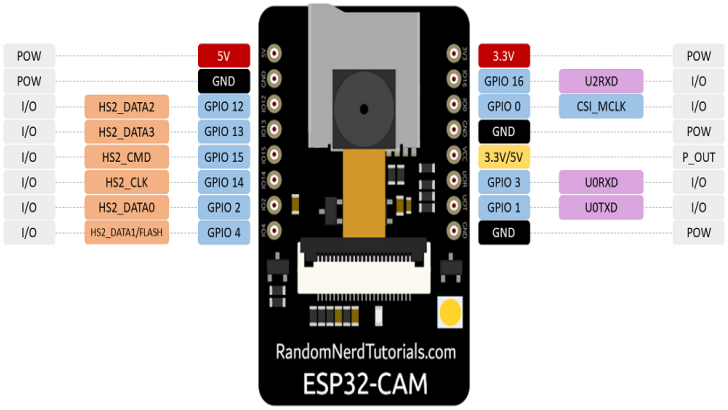
Pro splnění stanovených cílů byly zvoleny metody zahrnující výzkum dostupných technologií, vývoj mobilní aplikace v prostředí Android Studio a implementaci hardwarového řešení s využitím ESP32. Součástí procesu je testování systému, které ověří jeho funkčnost, kvalitu přenosu dat a stabilitu komunikace. Výběr těchto metod byl motivován snahou dosáhnout spolehlivého a efektivního řešení, které bude zároveň technologicky nenáročné.

Tento projekt se snaží propojit teoretické znalosti s praktickým řešením a nabídnout komplexní přístup k problematice ovládání IoT zařízení. V následujících kapitolách se text zaměří na samotnou implementaci, testování a hodnocení dosažených výsledků.

2 VYUŽITÉ TECHNOLOGIE

2.1 Hardware

2.1.1 ESP32-CAM



Obrázek č. 1 – popis jednotlivých pinů esp32

obrázek č. 2 – ESP32-cam

ESP32-CAM je mikrokontroler s kamerovým modulem, který umožňuje bezdrátovou komunikaci přes Wi-Fi a přenos obrazu v reálném čase. Díky dvoujádrovému procesoru a širokým možnostem připojení k periferiím je vhodný pro tento projekt, kde slouží k přenosu videa a ovládání auta.

2.1.2 H-bridge L298N

H-Bridge je obvod pro řízení směru otáčení motorů. Umožňuje pohyb auta vpřed i vzad tím, že přepíná polaritu připojení motoru. V projektu je použit pro efektivní řízení směru otáčení kol modelu auta.

2.1.3 Motory

Pro pohon auta byly použity dva stejnosměrné motory, které umožňují pohyb vpřed i vzad.

2.2 Software

2.2.1 Arduino IDE

Arduino IDE je oblíbené vývojové prostředí pro programování mikrokontrolerů, včetně ESP32. Umožňuje snadné psaní, kompilaci a nahrávání kódu. V tomto projektu bylo použito pro programování ESP32-CAM, poskytující podporu pro knihovny a nástroje pro bezdrátovou komunikaci a ovládání motorů.

2.2.2 Android studio

Android Studio je oficiální IDE pro vývoj Android aplikací. Bylo použito k vývoji mobilní aplikace pro ovládání auta, zahrnující návrh uživatelského rozhraní a komunikaci s ESP32-CAM. Nabízí nástroje pro testování a ladění aplikací na různých zařízeních.

2.2.3 Inventor

Autodesk Inventor byl využit k tvorbě 3D modelů hardwarových komponent potřebných pro projekt. Tento CAD software poskytuje nástroje pro parametrické modelování, simulace a tvorbu technické dokumentace. Inventor byl klíčový pro přesné navrhování komponent.

2.2.4 Prusa Slicer

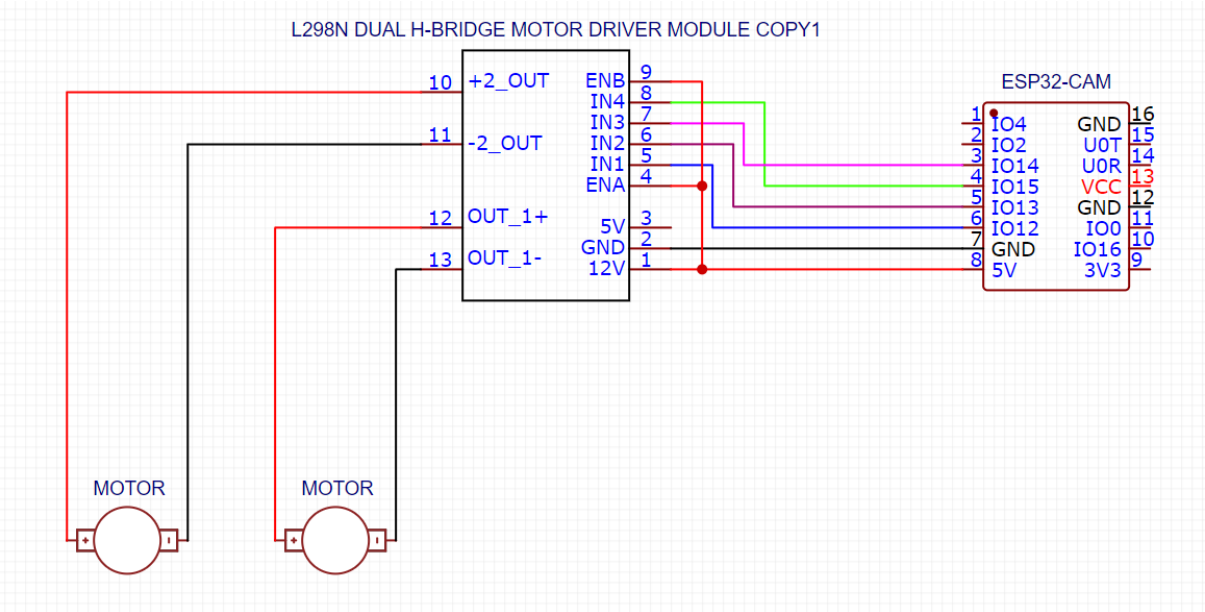
PrusaSlicer je pokročilý software pro přípravu modelů k 3D tisku. Tento nástroj byl využit k optimalizaci 3D modelů vytvořených v CAD softwarech.

3 ZAPOJENÍ HARDWARU

3.1 Schéma a popis zapojení

V tomto projektu je zapojení hardware zaměřeno na propojení mikrokontroleru ESP32-CAM**,** H-Bridge pro ovládání motorů a dalších součástek, jako jsou motory a případné senzory. Cílem je zajistit, aby všechny komponenty správně komunikovaly a umožnily dálkové ovládání modelu auta.

Obrázek č. 3 schéma zapojení



3.1.1 ESP32-CAM

Tento mikrokontroler slouží jako centrální jednotka pro ovládání celého systému. Má vestavěnou kameru, která snímá obraz a přenáší ho na mobilní zařízení prostřednictvím Wi-Fi. Dále ESP32-CAM řídí ovládání motorů a přijímá příkazy z Android aplikace.

3.1.2 H-Bridge

Tento obvod je zodpovědný za řízení směru a rychlosti otáčení motorů. H-Bridge je připojen k výstupním pinům ESP32-CAM a umožňuje přepínání polarity motorů pro pohyb vpřed nebo vzad.

3.1.3 Motory

Motory jsou připojeny k výstupům H-Bridge a zajišťují pohyb modelu auta. Řídí se směrem (vpřed, vzad) a rychlostí pomocí PWM signálů

3.2 Napájení

Napájení celého systému je zajištěno prostřednictvím powerbanky, která je pevně připevněna ke konstrukci autíčka. Tento způsob napájení byl zvolen pro svou mobilitu a snadnou dostupnost, což umožňuje autíčku volný pohyb bez potřeby externího napájecího zdroje. Powerbanka poskytuje dostatečný výkon pro provoz všech elektronických komponent, včetně ESP32, motorů a dalších připojených zařízení. Stabilní upevnění na konstrukci zajišťuje, že nedochází k nechtěnému pohybu nebo odpojení během provozu.

4 APLIKACE

4.1 Získávání videa

4.1.1 Video

Aplikace zajišťuje přístup k video streamu z kamery připojené k zařízení ESP32 prostřednictvím technologie WebView integrované do aplikace. WebView umožňuje načíst a zobrazit obsah z URL adresy generované na základě uživatelem zadané IP adresy.

4.1.2 Problém využitého řešení

Během implementace této funkce se objevil významný problém spojený s novými bezpečnostními pravidly Androidu. Tyto pravidla zakazují přístup k webovým zabezpečeného protokolu https. Jelikož webserver ESP32 nepodporuje protokol https, bylo nutné explicitně povolit přístup k nezabezpečeným stránkám.

Pro zajištění kompatibility s různými IP adresami jsem musel povolit přístup k **v**eškerým nezabezpečeným webům namísto omezení pouze na konkrétní URL. Tento krok představuje významné bezpečnostní riziko, jelikož aplikace umožňuje přístup k jakýmkoliv nezabezpečeným stránkám v síti.

4.2 Komunikace s ESP32

Pro komunikaci aplikace směrem k ESP32, například pro ovládání pomocí tlačítek, jsem využil technologii WebSocket. WebSocket server běží na zařízení ESP32 souběžně s webovým serverem. Když tedy uživatel stiskne tlačítko v aplikaci, příkaz je odeslán prostřednictvím WebSocketu na ESP32. Tento příkaz je následně zpracován na straně ESP32, kde může například aktivovat motory, změnit stav zařízení nebo provést jinou akci podle specifikace příkazu.

```
btnUp.setOnTouchListener((v, event) -> {
    switch (event.getAction()) {
        case MotionEvent.ACTION_DOWN:
            sendCommand("forward");
             break;
        case MotionEvent.ACTION_UP:
            sendCommand("stop");
              break;
   }
    v.performClick();
    return true;
});
```

- ukázka kódu pro zaslání WebSocket příkazu z aplikace

Komunikace přes WebSocket umožňuje obousměrnou výměnu informací. To znamená, že aplikace může nejen odesílat příkazy na ESP32, ale také může přijímat zpětné informace nebo odpovědi z ESP32. Tyto odpovědi mohou být využity pro zobrazení stavu zařízení, potvrzení přijatých příkazů nebo jiné akce, které umožňují aplikaci reagovat na chování zařízení. Tímto způsobem by šlo projekt rozšířit o interaktivní funkcionalitu, která umožní jak ovládání, tak i zpětnou vazbu mezi aplikací a ESP32. Využití oboustranné komunikace by také mohlo být možné vylepšení projektu do budoucna

4.3 Uživatelský vstup

Pro zajištění flexibility a přístupnosti je aplikace navržena tak, aby umožňovala uživatelům zadat IP adresu zařízení ESP32 manuálně. Tím se aplikace stává univerzální a snadno použitelnou i v případech, kdy je zařízení ESP32 nasazeno v různých sítích nebo prostředích.

4.3.1 Zadání IP adresy

Aplikace obsahuje dialogové okno pro zadání nebo úpravu IP adresy, které je vyvoláno vždy při prvním zapnutí aplikace, jelikož je na IP adrese závislá celá funkčnost aplikace, následně pak může uživatel vyvolat toto dialogové okno sám v nastavení aplikace a změnit tak IP adresu za běhu aplikace

4.3.2 Kontrola

Po zadání IP adresy aplikace ověřuje, zda odpovídá standardnímu formátu IPv4. Tato validace je zajištěna pomocí regulárního výrazu v metodě `validateIpAddress ().`

Regulární výraz kontroluje, zda se zadaná adresa skládá ze čtyř čísel v rozsahu 0–255 oddělených tečkou. Pokud zadaná IP adresa neodpovídá tomuto formátu, aplikace zobrazí chybové hlášení a umožní uživateli adresu opravit.

```
private boolean validateIpAddress(String ip) {
    String ipPattern = "^((25[0-5]|2[0-4][0-9]|[01]?[0-9][0-
9]?)\\.){3}(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)$";
    return ip.matches(ipPattern);
}
```

5 ŘÍZENÍ AUTA – KÓD ESP

5.1 Připojení k síti

Připojení k síti je klíčovou součástí projektu, jelikož je celá funkčnost systému závislá na správném připojení zařízení ESP32 k síti. ESP32 se připojuje k WiFi síti pomocí zadaného SSID (název sítě) a hesla. Tyto hodnoty jsou uloženy v samostatném souboru pro zvýšení bezpečnosti, aby se citlivé informace nezobrazovaly přímo v hlavním kódu. Takto je zajištěno, že připojení k síti je možné pouze tehdy, když jsou správné autentizační údaje, což pomáhá chránit zařízení před neoprávněným přístupem.

Při spuštění zařízení se ESP32 pokusí připojit k síti pomocí těchto údajů. Pokud je připojení úspěšné, zařízení získá přístup k internetu a může zahájit komunikaci s dalšími zařízeními, jako je například Android aplikace. V případě, že připojení selže, zařízení se pokusí znovu připojit, dokud nebude úspěšné, čímž je zajištěna stabilita projektu v různých síťových podmínkách.

5.2 Web server pro streamování videa

Výstup z kamery připojené k zařízení ESP32 je streamován prostřednictvím webového serveru, který je vytvořen pomocí knihovny **esp_http_server**. Tento server poskytuje video stream ve formátu MJPEG, který je dostupný pro uživatele prostřednictvím webového prohlížeče.

Po připojení zařízení ESP32 k Wi-Fi síti, je automaticky přidělena IP adresa, kterou zařízení používá pro komunikaci s ostatními zařízeními v síti. Tato IP adresa je následně použita pro přístup k webovému serveru ESP32 a tím i k živému video streamu z kamery. Uživatel tak může jednoduše přistupovat k videu přes jakýkoliv prohlížeč nebo aplikaci, která je schopná načíst URL adresu streamu.

Tento přístup k video streamu je klíčovou součástí projektu, jelikož umožňuje sledování obrazu z kamery v reálném čase, což je základem pro ovládání zařízení a interakci s uživatelem v aplikaci.

5.3 Web Socket

Souběžně s webovým serverem, který streamuje video, běží na zařízení ESP32 i WebSocket server. Tento server přijímá příkazy z mobilní aplikace a na základě těchto příkazů spouští další ovládací funkce na straně ESP32. WebSocket server je klíčový pro komunikaci mezi aplikací a zařízením, což umožňuje ovládání pohybu zařízení a další akce.

Zatímco webový server pro video stream běží na standardním portu (80), WebSocket server běží na portu 81. Oba servery používají stejnou IP adresu zařízení ESP32, což umožňuje aplikaci komunikovat s oběma službami současně. WebSocket server je navržen tak, aby umožnil efektivní výměnu dat mezi aplikací a ESP32 s nízkou latencí, což je zásadní pro reakce na uživatelské příkazy.

Tento dvojitý serverový přístup na stejné IP adrese zajišťuje efektivní a plynulou komunikaci mezi aplikací a zařízením, přičemž video stream a ovládání probíhá nezávisle na sobě, což zajišťuje plynulý zážitek pro uživatele.

```
if (message == "forward") {
  forward();
}
else if (message == "backward") {
  backward();¨
}
else if (message == "left") {
   turnLeft();
}
else if (message == "right") {
   turnRight();
}
else if(message == "led"){
   if(isOn == false){
      digitalWrite(LED_GPIO_NUM, HIGH);
        isOn = true;
    }
    else{
       digitalWrite(LED_GPIO_NUM, LOW);
        isOn = false;
    }
}
else if(message == "stop"){
    stopMotors();
}
else {
    Serial.println("unknow command");
}
```

- Ukázka kódu pro zpracování příchozích WebSocket příkazů z aplikace

5.4 Ovládání motorů

Pro ovládání motorů jsou napsány čtyři funkce, které umožňují ovládat pohyb zařízení v různých směrech. Tyto funkce jsou navrženy tak, aby poskytovaly základní pohybové příkazy pro zařízení, a to, jak pro pohyb vpřed, vzad, tak pro otáčení na obě strany. Jednotlivé funkce jsou spouštěny podle příkazů obdržených z aplikace pomocí WebSocket.

```
void forward() {
    digitalWrite(IN1, LOW);   // Motor 1 backward
    digitalWrite(IN2, HIGH);  // Motor 1 backward
    digitalWrite(IN3, LOW);   // Motor 2 backward
    digitalWrite(IN4, HIGH);  // Motor 2 backward
}

void turnRight() {
    digitalWrite(IN1, LOW);   // Motor 1 backward
    digitalWrite(IN2, HIGH);  // Motor 1 backward
    digitalWrite(IN3, HIGH);  // Motor 2 forward
    digitalWrite(IN4, LOW);   // Motor 2 forward
}
```

- Ukázka kódu pro řízení motorů vpřed a zatočení doprava

6 MODEL A KONSTRUKCE DRŽÁKU ESP

6.1 Modely

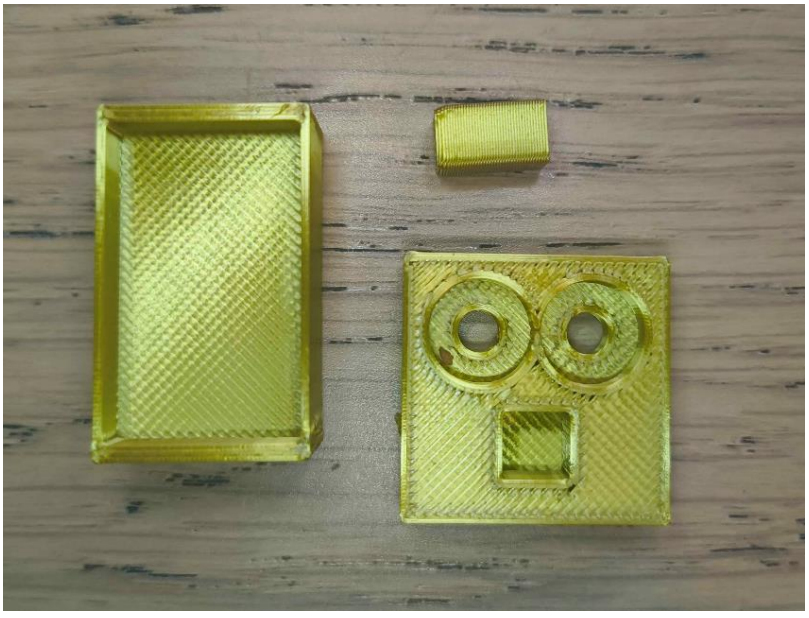
Držák pro zařízení ESP32 je navržen jako sestava tří samostatných modelů, které byly následně vytisknuty na 3D tiskárně Prusa Mini. Každý z těchto dílů má specifickou funkci a společně tvoří pevnou a stabilní konstrukci pro upevnění ESP32 ke konstrukci autíčka.

První díl je určen pro upevnění celé konstrukce k desce automobilu. Tento díl je navržen tak, aby pevně držel celý systém na místě, čímž zajišťuje stabilitu a bezpečnost během používání zařízení. Díky přesně navrženým otvorům pro šrouby a montážní body se tento díl snadno připevní na požadované místo.

Druhý díl slouží k propojení všech částí konstrukce dohromady. Tento díl má úkol spojit jednotlivé segmenty držáku, čímž vytváří kompaktní celek.

Třetí díl je určen pro samotné držení zařízení ESP32. Tento díl je pečlivě navržen tak, aby bezpečně a stabilně udržel ESP32, zároveň umožňuje snadnou montáž a demontáž zařízení. Držák má dostatečnou prostorovou toleranci pro zařízení, čímž zajišťuje bezpečné uchycení i v případě vibrací nebo otřesů během pohybu auta.

Obrázek č.4 vytištěné díly



6.2 Konstrukce

Ke konstrukci celého držáku je zapotřebí ještě dvou vrutů, které jsou použity k upevnění držáku k desce automobilu. Pro tento účel jsem vybral vruty typu PH 4,0x16 s půlkulatou hlavou, které odpovídají konstrukci modelu držáku.

Vruty slouží k připevnění spodní části držáku ke zbytku automobilu. Výběr vrutů byl zvolen právě kvůli jejich snadné odnímatelnosti, což umožňuje flexibilitu při manipulaci s držákem a připojeným zařízením. Díky tomu je možné držák snadno připevnit nebo demontovat, aniž by bylo nutné provádět složité zásahy do konstrukce auta.

Po připevnění spodního dílu držáku vruty k automobilu se na něj jednoduše nasune spojující díl. Tento díl má za úkol propojit jednotlivé části držáku a poskytnout stabilní podporu. Na druhé straně spojujícího dílu se následně nasadí poslední díl, do kterého je bezpečně umístěno zařízení ESP32. Tento proces je velmi jednoduchý a rychlý, což usnadňuje montáž celého systému.

Tímto způsobem je zařízení ESP32 pevně uchyceno a stabilně umístěno v držáku, což zajišťuje, že je zařízení bezpečně připojeno k automobilu a připraveno k použití. Celková konstrukce je navržena tak, aby byla co nejefektivnější, snadno manipulovatelná a zároveň pevná a spolehlivá během používání.

obrázek č.5 esp ve zkonstruovaném držáku

ZÁVĚR

V rámci této práce byl navržen a realizován systém pro ovládání a monitorování zařízení založeného na platformě ESP32 pomocí mobilní aplikace. Hlavním cílem bylo vytvoření funkční aplikace, která umožňuje komunikaci s ESP32 přes Wi-Fi síť, streamování videa v reálném čase a ovládání zařízení prostřednictvím WebSocketu. Cílem bylo rovněž navrhnout intuitivní uživatelské rozhraní, které umožní snadné ovládání autíčka a zároveň poskytne zpětnou vazbu o stavu zařízení.

Výsledky práce ukazují, že navržený systém je plně funkční a splňuje stanovené cíle. Aplikace úspěšně zajišťuje streamování videa a komunikaci mezi zařízením a mobilní aplikací. Uživatelé mohou snadno ovládat zařízení, přičemž aplikace podporuje flexibilní zadání IP adresy zařízení, což umožňuje její univerzálnost v různých prostředích.

V praxi lze tento systém uplatnit v různých oblastech, například v dálkovém ovládání robotických zařízení, sledování pohybu v reálném čase. V budoucnu by mohl být tento systém dále vylepšen, například použitím kvalitnější kamery pro lepší rozlišení obrazu nebo zajištěním přenosu videa s nižší latencí, což by umožnilo plynulejší a rychlejší interakci mezi uživatelem a zařízením.

Dalším možným vylepšením by byla integrace ovládání rychlosti a senzorů pro přidání autonomního režimu řízení autíčka. Senzory by mohly například detekovat překážky a automaticky upravovat směr pohybu. V rámci vylepšení by také bylo možné integrovat obousměrnou komunikaci a posílat tak do aplikace zpětnou vazbu o stavu autíčka, jako je například úroveň baterie nebo aktuální rychlost, a tyto informace přenášet do aplikace pro lepší kontrolu nad zařízením.

Tato práce ukazuje, jak kombinace hardwaru a softwaru může přinést efektivní a inovativní řešení pro mobilní aplikace, které komunikují se zařízeními v reálném čase. Další vylepšení a rozšíření tohoto systému by mohlo otevřít nové možnosti pro různé aplikace v oblasti robotiky a automatizace.

GitHub: https://github.com/FabianPavel/Final-Project

SEZNAM POUŽITÝCH INFORMAČNÍCH ZDROJŮ

[1] SATTLER, MARKUS. *WebSockets for Arduino (Server + Client)*. Online. 2024.https://github.com/Links2004/arduinoWebSockets/blob/master/README.md . [cit. 2024-12-18].

[2] *Connect 3D Printed Parts | Design for Mass Production 3D Printing* [@Slant 3D]. Online. 2024. Dostupné z: YouTube, https://www.youtube.com/watch?v=djm5tCFn9S0. [cit. 2024-12-26].

[3] JETBRAINS. *Android codelabs*. Online. 2024. Dostupné z: https://developer.android.com/get-started/codelabs. [cit. 2025-01-09]

[4] GITHUB. *Esp32 https server*. Online. 2020. Dostupné z: https://github.com/fhessel/esp32_https_server. [cit. 2024-09-06].

[5] RANDOM NERD TUTORIALS. *ESP32-cam pinout*. Online. 2021. Dostupné z: https://randomnerdtutorials.com/esp32-cam-ai-thinker-pinout/. [cit. 2024-09-06].