



PLÁN [OBNOVY]

Nitrianske kompetenčné centrum kybernetickej bezpečnosti
Fakulta prírodných vied a informatiky
Univerzita Konštantína Filozofa v Nitre
Tr. A. Hlinku 1, 949 01 Nitra

DATABÁZY A ICH VYUŽITIE V OBLASTI MIKROKONTROLÉROV S DÔRAZOM NA BEZPEČNOSŤ V IOT

Predmet: 24 – Internet vecí

Téma

Databázy zohrávajú kľúčovú úlohu v oblasti mikrokontrolérov a internetu vecí (IoT), najmä pri správe, ukladaní a analýze veľkého množstva dát generovaných senzormi a zariadeniami. V prostredí IoT sú mikrokontroléry často zodpovedné za zber údajov z okolia (napr. teplota, vlhkosť, pohyb) a ich následné odosielanie do centrálnej databázy – buď lokálnej (napr. SQLite, TinyDB) alebo cloudovej (napr. Firebase, AWS DynamoDB).

Bezpečnosť je v tomto kontexte mimoriadne dôležitá. Prenos dát medzi mikrokontrolérom a databázou musí byť šifrovaný (napr. pomocou TLS), aby sa predišlo úniku citlivých informácií. Okrem

toho je dôležité zabezpečiť autentifikáciu zariadení, kontrolu prístupu k databáze a pravidelné aktualizácie firmvéru, aby sa minimalizovalo riziko zneužitia.

V praxi sa často využívajú ľahké databázové riešenia optimalizované pre nízku spotrebu energie a obmedzené výpočtové zdroje, čo je typické pre mikrokontroléry. V kombinácii s bezpečnostnými protokolmi a cloudovou infraštruktúrou tak databázy umožňujú efektívne a bezpečné riadenie IoT systémov.

autor
skoprda@ukf.sk

Obsah

Abstrakt	3
Cieľová skupina.....	3
Požiadavky	3
Rozsah a formát.....	3
 OBSAH	 4

Abstrakt

Rastúci význam internetu vecí (IoT) prináša nové výzvy v oblasti spracovania a zabezpečenia dát generovaných mikrokontrolérmi. Tieto zariadenia, často nasadzované v prostrediach s obmedzenými výpočtovými a pamäťovými zdrojmi, zohrávajú kľúčovú úlohu pri zbere a prenose údajov do databázových systémov. Cieľom tejto práce je preskúmať možnosti integrácie databáz do IoT architektúr s dôrazom na efektívne a bezpečné ukladanie dát. Analyzujú sa rôzne typy databáz vhodné pre mikrokontroléry (napr. embedded databázy, cloudové riešenia) a diskutujú sa bezpečnostné mechanizmy ako šifrovanie, autentifikácia a kontrola prístupu. Výsledkom je návrh odporúčaní pre vývojárov IoT riešení, ktoré zohľadňujú nielen technické obmedzenia zariadení, ale aj rastúce požiadavky na ochranu dát a súkromia.

Autor: Štefan Koprda

Email: skoprda@ukf.sk

Univerzita: Fakulta prírodných vied a informatiky, Univerzita Konštantína Filozofa v Nitre

Cieľová skupina

Cieľovou skupinou sú študenti aplikovanej informatiky.

Vzdelávacie ciele

- Účastníkom osvojované vedomosti a zručnosti
- Účastníkom rozvíjané spôsobilosti

Východiskové predpoklady a požiadavky na účastníkov

- Práca s databázami, python, počítačové siete

Použité metódy

- Práca v skupinách

Požiadavky

Prístrojové vybavenie

- Počítače, sada mikrokontrolérov Arduino, arduinoide

Priestorové požiadavky

- Učebňa - laboratórium

Veľkosť skupiny

- Maximálne 10 študentov

Rozsah a formát

- Trvanie: 3 hodiny
- Forma: prezenčne

Obsah

Úvod

V ére prepojených zariadení a inteligentných technológií sa čoraz viac projektov sústreďuje na bezdrôtovú komunikáciu a možnosť vzdialeného ovládania alebo zberu dát cez internet. Tradičné vývojové dosky, ako je Arduino Uno, síce ponúkajú jednoduchý spôsob, ako ovládať rôzne komponenty, no pre projekty v oblasti internetu vecí (IoT) im chýba natívna schopnosť pripojenia na WiFi sieť. Práve túto potrebu rieši Arduino Uno WiFi Rev2 – modernizovaná verzia klasického Uno, ktorá kombinuje osvedčenú jednoduchosť s možnosťou bezdrôtovej konektivity.

Arduino Uno WiFi Rev2 vzniklo ako odpoveď na požiadavku používateľov mať k dispozícii spoľahlivú, výkonnú a kompaktnú dosku, ktorá bude schopná pripojiť sa k internetu bez potreby externých modulov. V porovnaní s pôvodným modelom prináša viacero vylepšení – výkonný mikrokontrolér ATmega4809, integrovaný WiFi čip od firmy u-blox (NINA-W102), vstavaný kryptografický čip pre bezpečné pripojenie a možnosť programovania cez Arduino IDE rovnako jednoducho ako pri predchádzajúcich verziách.

Doska je vhodná pre všetkých, ktorí chcú budovať IoT zariadenia, diaľkovo monitorovať senzory, ovládať osvetlenie alebo spotrebiče cez internet, alebo vytvárať interaktívne systémy, ktoré komunikujú s cloudom, databázami či mobilnými aplikáciami. Vďaka svojej kompatibilite s Arduino ekosystémom, podporným knižnicami a komunitou, je Arduino Uno WiFi Rev2 výbornou voľbou pre vzdelávanie, prototypovanie aj praktické aplikácie v oblasti automatizácie a diaľkového riadenia.



Prehľad pinov Arduino Uno WiFi Rev2

Arduino Uno WiFi Rev2 má rovnaké základné rozloženie pinov ako klasické Arduino Uno, čo zaručuje kompatibilitu s väčšinou shieldov a rozširujúcich modulov. Celkovo obsahuje **14 digitálnych pinov**, **6 analógových vstupov**, napájacie piny a špecializované piny pre komunikáciu.

1. Digitálne piny (D0 – D13)

Označené ako **D0 až D13** (na okraji dosky).

Môžu slúžiť ako vstupy alebo výstupy (pomocou `pinMode()`, `digitalWrite()`, `digitalRead()`).

Niektoré z nich majú špeciálne funkcie:

Pin	Funkcia
D0 (RX)	Prijíma sériový signál (UART)
D1 (TX)	Posiela sériový signál (UART)
D2 – D13	Bežné digitálne I/O
D3, D5, D6, D9, D10, D11	Podpora PWM výstupu (<code>analogWrite()</code>) – označené vlnovkou (~)

2. Analógové vstupy (A0 – A5)

Umožňujú čítanie analógových hodnôt (0–1023).

Používajú sa najmä na pripojenie senzorov (napr. teploty, svetla, vlhkosti).

Okrem analógového čítania sa dajú použiť aj ako digitálne piny (A0 = D14, atď.).

3. Napájacie piny

Pin	Popis
VIN	Vstupné napätie pri napájaní zvonka (7–12V)
5V	Výstup 5V regulovaného napätia
3.3V	Výstup 3.3V (obmedzený prúd, max. ~50 mA)
GND	Zem (spoločný záporný pól) – viaceré piny
RESET	Reštartuje mikrokontrolér (aktívny LOW)

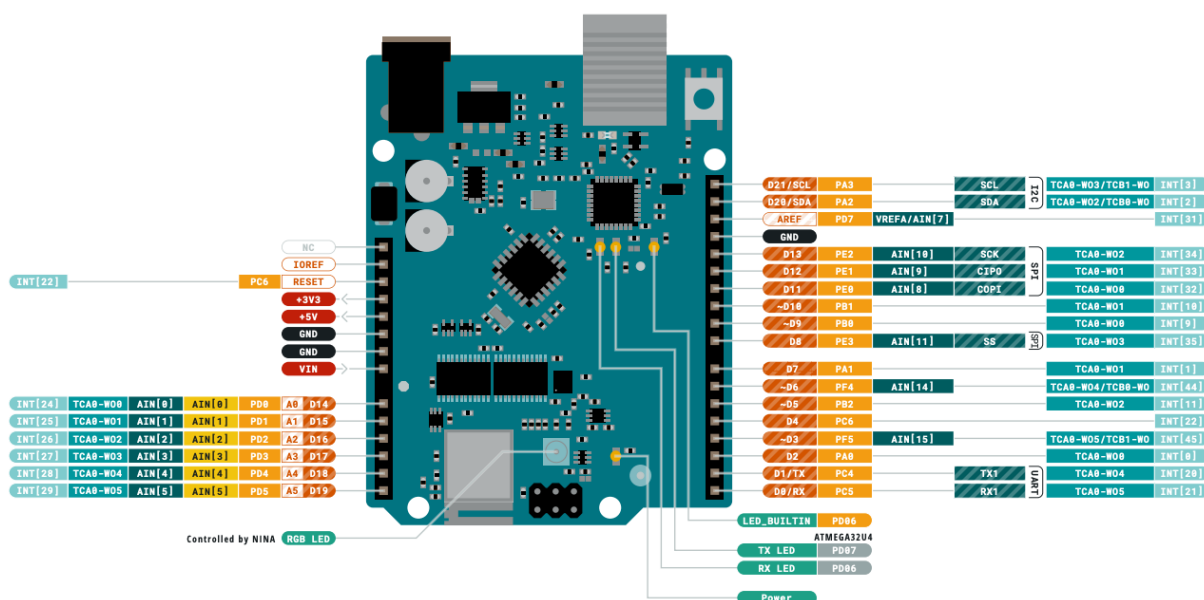
4. Špeciálne komunikačné piny

Protokol	Piny	Funkcia
UART	D0 (RX), D1 (TX)	Sériová komunikácia
I2C	A4 (SDA), A5 (SCL)	Komunikácia s I2C zariadeniami (napr. senzory, displeje)
SPI	D10 (SS), D11 (MOSI), D12 (MISO), D13 (SCK)	Vysokorychlostná komunikácia (napr. s WiFi modulom)

Poznámka: Na Uno WiFi Rev2 je I2C riadené interným čipom (nie ATmega328P, ale ATmega4809), takže sa technicky mapuje trochu odlišne ako na klasickom Uno, ale zvonka je rozhranie kompatibilné.

5. Iné piny a vlastnosti

- LED_BUILTIN (D13): Na tomto pine je pripojená vstavaná LED – používa sa na základné testovanie.
- ICSP Header: Šesťpinový konektor na programovanie cez SPI (napr. externým programátorom).
- WiFi LED: Indikátor stavu WiFi modulu.
- Crypto čip: K doske patrí aj čip ATECC608A, ktorý je napojený cez I2C a slúži na bezpečné ukladanie kľúčov a šifrovanie.



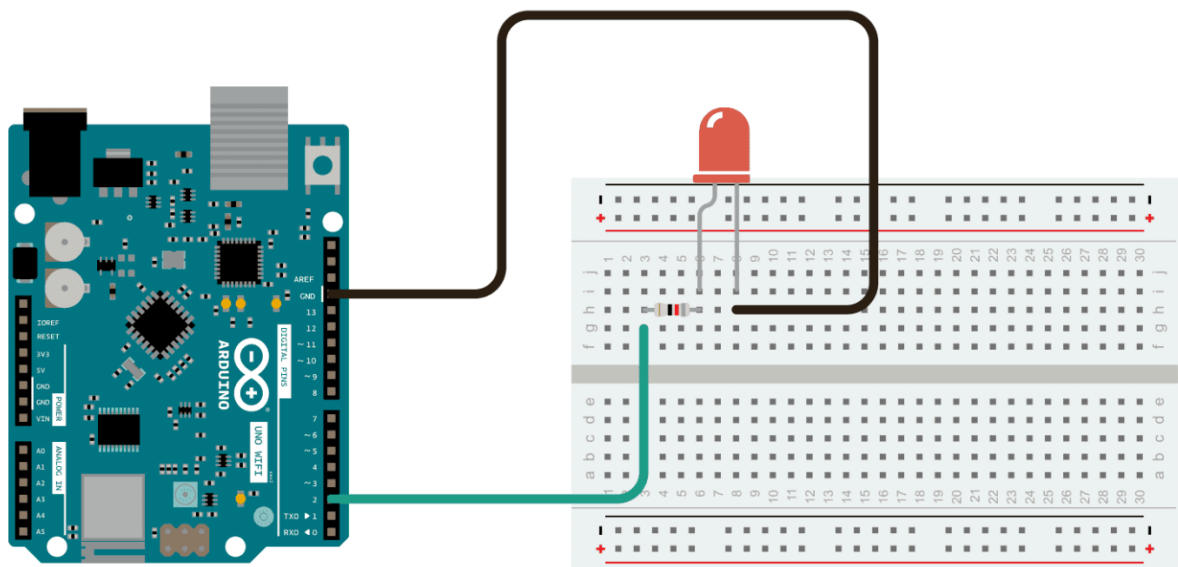
Jednoduchý príklad zapojenia

Základné zapojenie: Blikanie LED diódou

Potrebné komponenty:

- 1 × Arduino Uno WiFi Rev2
- 1 × LED dióda (napr. červená)
- 1 × rezistor 220–330 Ω
- 1 × nepájivé pole (breadboard)
- 2 × prepojovacie vodiče

Schéma zapojenia:



Postup zapojenia:

1. Pripojte **anódu** (dlhšia nožička) LED diódy k jednému koncu rezistora.
2. Druhý koniec rezistora spojte s digitálnym pinom **D2** na Arduino doske.
3. Pripojte **katódu** (kratšia nožička) LED diódy k **GND** (uzemnenie) na Arduino.

Ukázkový kód:

```
void setup() {
  pinMode(2, OUTPUT); // Nastaví pin D2 ako výstup
}

void loop() {
  digitalWrite(2, HIGH); // Zapne LED
  delay(1000);           // Čaká 1 sekundu
  digitalWrite(2, LOW);  // Vypne LED
  delay(1000);           // Čaká 1 sekundu
}
```

Inštrukcie:

- Otvorte Arduino IDE.
- Vložte vyššie uvedený kód.
- Vyberte správnu dosku: **Arduino Uno WiFi Rev2**.
- Pripojte Arduino k počítaču cez USB a nahrajte kód. [Arduino Documentation](#)

Po nahratí by mala LED dióda blikáť v intervale 1 sekundy.

Tento jednoduchý projekt je skvelým úvodom do práce s Arduino Uno WiFi Rev2 a poskytuje základ pre ďalšie experimenty, ako je ovládanie viacerých LED diód alebo senzorov.

Arduino a Firebase + vytvorenie vlastnej databázy

Firebase je platforma od spoločnosti Google, ktorá poskytuje rôzne nástroje a služby pre vývoj webových a mobilných aplikácií. Jednou z jej najdôležitejších súčastí je **Firebase databáza** – cloudová databáza, ktorá umožňuje **ukladanie, synchronizáciu a zdieľanie dát v reálnom čase**. Firebase databáza je ideálna najmä pre aplikácie, ktoré vyžadujú rýchlu komunikáciu medzi zariadeniami a aktuálne údaje pre všetkých používateľov súčasne.

Firebase ponúka dva typy databáz:

1. **Realtime Database** – hierarchicky štruktúrovaná databáza typu NoSQL, ktorá funguje na princípe stromu (JSON formát). Je vhodná pre rýchlu synchronizáciu dát medzi klientmi v reálnom čase.
2. **Cloud Firestore** – novšia, flexibilnejšia databáza, ktorá podporuje štruktúrovanejšie údaje (kolekcie a dokumenty), silnejšie dotazovanie a lepšiu integráciu s ďalšími Google službami.

Veľkou výhodou Firebase databázy je, že funguje **v cloude** – nie je potrebné spravovať vlastný server. Dáta sa dajú čítať a zapisovať priamo z mobilnej, webovej alebo IoT aplikácie. Prístup sa riadi pomocou **prístupových pravidiel a autentifikácie**, takže vývojár môže presne určiť, kto čo môže vidieť alebo meniť.

V kontexte **mikrokontrolérov**, ako je **Arduino alebo ESP32**, je Firebase čoraz populárnejšou voľbou, pretože umožňuje jednoduché **odosielanie a prijímanie dát cez WiFi** – napríklad sledovanie teploty, vlhkosti, stavu dverí či svetla v reálnom čase cez mobilnú aplikáciu alebo webové rozhranie. Firebase tak predstavuje moderné, jednoduché a výkonné riešenie na správu dát pre IoT projekty, ktoré potrebujú byť pripojené k internetu.



Základné vlastnosti Firebase databázy

Firebase databáza (Realtime Database aj Cloud Firestore) je navrhnutá tak, aby poskytovala **rýchlu, spoľahlivú a jednoducho použiteľnú platformu pre ukladanie dát v reálnom čase**, pričom odstraňuje potrebu budovania a správy vlastného serverového zázemia. Tu sú jej kľúčové vlastnosti:

1. Ukladanie dát v cloude

Firebase je **plne cloudová služba**, čo znamená, že všetky údaje sa ukladajú na serveroch Googlu. Nie je potrebné nastavovať vlastný server, databázový engine, ani sa starať o zálohovanie – všetko spravuje Firebase automaticky.

2. Synchronizácia dát v reálnom čase

Jednou z najsilnejších vlastností Firebase je schopnosť **synchronizovať dáta v reálnom čase** medzi viacerými klientmi. Ak sa dáta na jednom zariadení zmenia, okamžite sa aktualizujú aj na všetkých ostatných pripojených zariadeniach. Táto vlastnosť je obzvlášť užitočná v:

- chatovacích aplikáciách,
- online spolupráci,
- IoT systémoch (napr. smart home),
- live notifikáciách a sledovaní stavov.

3. NoSQL štruktúra dát

Firebase databázy sú **nerelačné (NoSQL)**, čo znamená, že dáta nie sú uložené v tabuľkách, ale vo forme:

- **stromovej štruktúry JSON** (Realtime Database), alebo
 - **kolekcií a dokumentov** (Cloud Firestore).
- Tento prístup umožňuje vysokú flexibilitu a rýchlosť pri zápise aj čítaní údajov, no zároveň si vyžaduje premyslený návrh dátovej štruktúry.

4. Bezpečnosť cez pravidlá prístupu

Firebase databáza umožňuje definovať podrobné **bezpečnostné pravidlá**, ktoré určujú, kto môže čítať alebo zapisovať údaje. Tieto pravidlá môžu byť založené na:

- autentifikácii používateľa (napr. prihlásený vs. anonymný používateľ),
- štruktúre dát (napr. každý používateľ vidí len svoje údaje),
- logike aplikácie (napr. len admini môžu upravovať konfiguráciu).

5. Integrácia s ďalšími Firebase službami

Firebase databáza úzko spolupracuje s ostatnými nástrojmi Firebase, napríklad:

- **Firebase Authentication** – na overovanie používateľov,
- **Firebase Hosting** – na nasadenie webovej aplikácie,
- **Cloud Functions** – na spúšťanie kódu na serveri ako reakcia na zmeny v databáze,
- **Firebase Storage** – na ukladanie súborov (obrázkov, dokumentov),
- **Firebase Analytics** – na sledovanie správania používateľov.

6. Podpora viacerých platforiem

Firebase je **multiplatformový** – podporuje vývoj pre:

- Android,
- iOS,
- Web,
- C++,
- Unity,
- a dokonca aj prístup z **mikrokontrolérov** (napr. **ESP32, ESP8266**) cez REST API alebo Firebase Arduino knižnice.

7. Offline podpora

Firebase klienti majú možnosť pracovať aj **offline**. Dáta sa dočasne ukladajú lokálne a po opätovnom pripojení sa automaticky synchronizujú so serverom. Toto je veľmi výhodné pre aplikácie, ktoré potrebujú fungovať aj v prípade výpadku pripojenia.

8. REST API a SDK

Firebase databázu možno ovládať nielen pomocou knižníc pre konkrétne platformy, ale aj cez **REST API** – čo umožňuje prístup aj zo zariadení ako mikrokontroléry (napr. ESP32, ktoré posielajú dáta cez HTTP požiadavky).

9. Škálovateľnosť a výkonnosť

Firebase je postavený na infraštruktúre Googlu a bez problémov zvláda rastúce množstvo údajov aj používateľov. Je vhodný pre malé školské projekty aj veľké produkčné aplikácie.

10. Bezplatný a škálovateľný cenový model

Firebase ponúka **Free tier (Spark plan)** pre malé projekty s obmedzenými zdrojmi a možnosťou neskoršieho prechodu na **Platené plány (Blaze)** podľa objemu dát a operácií.

Vytvorenie vlastnej databázy

Tu je prehľadný krokový postup, ako si vytvoriť vlastnú Firebase databázu, vhodnú pre prácu s webom, mobilnou aplikáciou alebo napríklad aj s mikrokontrolérom.

Vytvorenie databázy:

1. Otvorte nasledujúci link: console.firebase.google.com
2. Prihláste sa Vaším gmail účtom.
3. Kliknite na "Create a project"
4. Vyberte názov projektu a kliknite na continue.
5. Zvolte, či chcete použiť AI assistenta a kliknite na continue.
6. Zvolte, či chcete použiť Google Analytics a kliknite na continue.

Teraz sa nám podarilo úspešne vytvoriť nový project. Na ľavo v menu vyberte "Build" a následne "Realtime Database".

1. Následne kliknite na "Create database".
2. Realtime Database location: United States
3. Zvolte možnosť "Start in test mode" a kliknite na Enable

Teraz sa nám podarilo úspešne vytvoriť novú real time databázu, do ktorej môžeme posilať dáta, napríklad z Arduina.

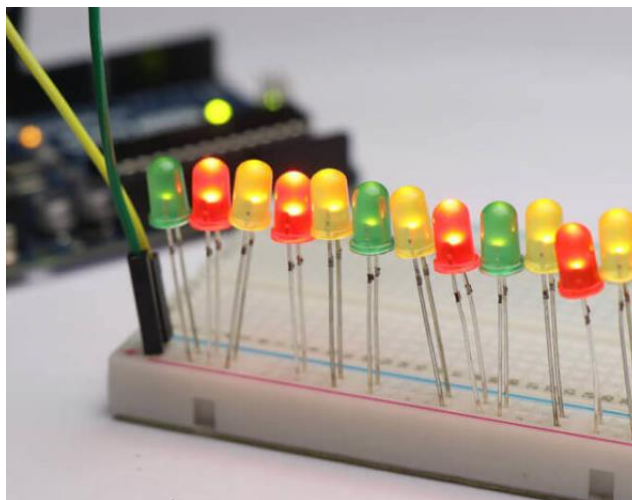
V prípade potreby si prosím pozrite nasledovný tutorial:



Úloha:

Ovládanie led diód cez webový server s využitím Firebase databázy

Toto zapojenie umožňuje ovládanie dvoch LED diód na doske Arduino UNO WiFi Rev 2 cez internet pomocou služby Firebase Realtime Database. LED diódy sú pripojené na digitálne piny a ich stav sa mení na základe údajov uložených v cloud. Pomocou jednoduchého webového rozhrania môže používateľ na diaľku zapínať alebo vypínať jednotlivé LED diódy, pričom Arduino pravidelne číta hodnoty z databázy a aktualizuje výstupy. Projekt demonštruje základnú prácu s IoT (Internet of Things), pripojením Wi-Fi a cloudovou databázou.



Hardvérové zapojenie

HARDVÉR

1. Potrebné komponenty:

- 1× Arduino UNO WiFi Rev 2
- 2× LED diódy (ľubovoľnej farby)
- 2× Rezistory 220–330 ohmov
- Prepojovacie vodiče (male-male)
- Breadboard (nepovinné, pre lepšiu organizáciu)

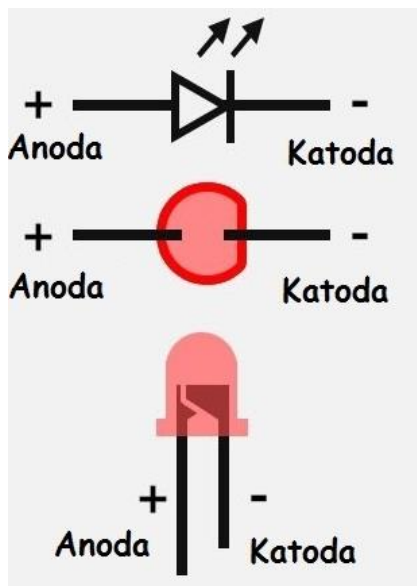
2. Zapojenie LED diód:

Arduino pin	Pripojenie
D5	Anóda LED1 cez rezistor do GND
D6	Anóda LED2 cez rezistor do GND
GND	Katódy oboch LED diód

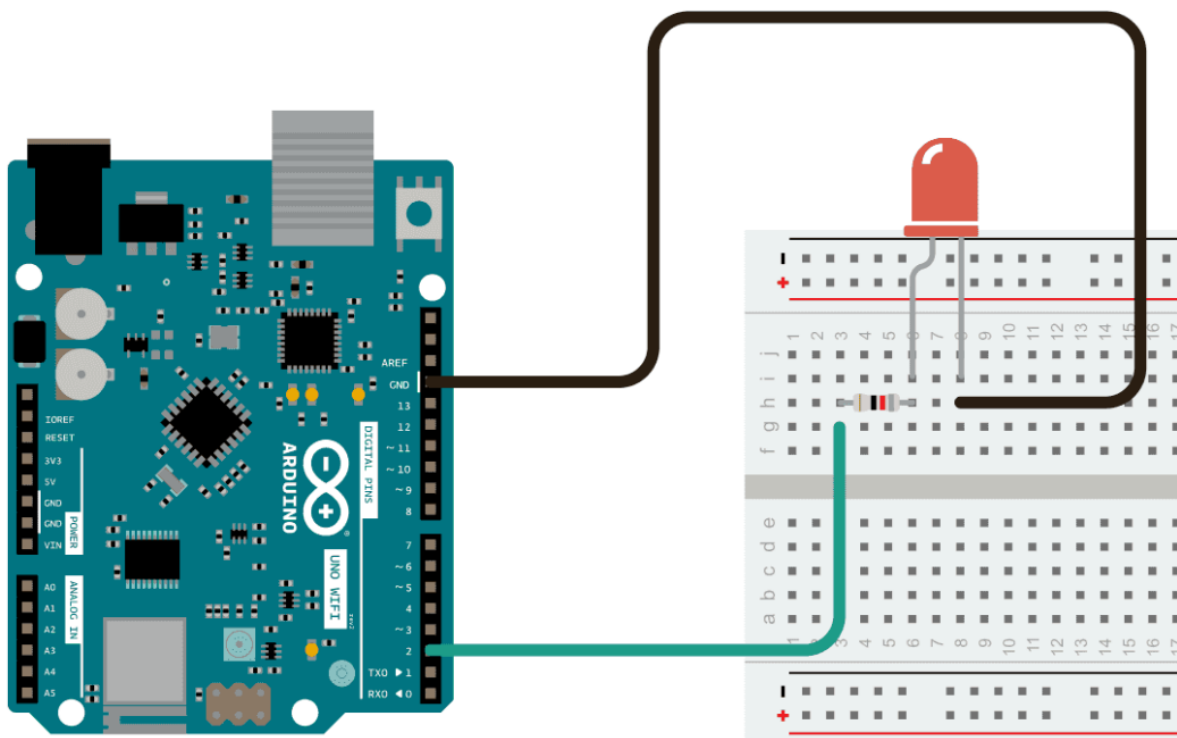
Detail:

- **LED1:** Pin D5 → rezistor (napr. 220 Ω) → anóda LED → katóda na GND
- **LED2:** Pin D6 → rezistor → anóda LED → katóda na GND

Poznámka: Rezistor môžeš zaradiť buď pred anódu alebo za katódu – zapojenie bude fungovať rovnako.



Príklad zapojenia



Inštalácia knižníc

V Arduino IDE je potrebné nainštalovať tieto knižnice (cez **Library Manager**):

1. **Wi-FiNINA**: Knižnica pre prácu s Wi-Fi modulom v doskách ako Arduino Uno WiFi Rev 2, umožňuje pripojenie k internetu.
2. **Firebase_Arduino_WiFiNINA**: Zabezpečuje komunikáciu s Firebase

Nastavenie databázy

Je potrebné upraviť bezpečnostné pravidlá v službe Firebase Realtime Database preto, aby Vaša Arduino doska mohla čítať a zapisovať údaje do databázy. V predvolenom nastavení je prístup obmedzený, aby bola databáza chránená pred neoprávneným prístupom.

Ak chcete počas vývoja jednoducho testovať funkčnosť komunikácie medzi webovým rozhraním, Firebase a hardvérom, odporúča sa dočasne povoliť úplný prístup (.read a .write nastaviť na true). Tým umožníte, aby Vaše zariadenie aj webová stránka mohli bez obmedzení pracovať s dátami v databáze.

Postup na úpravu bezpečnostných pravidiel v Firebase RTDB

1. **Firebase Console**
 - Prejdi na: <https://console.firebase.google.com>
 - Vyber svoj projekt.
2. **Choď do časti „Realtime Database“**
 - V ľavom menu klikni na „Build“ → „Realtime Database“.
 - Vyber svoju databázu.
3. **Klikni na kartu „Rules“ (Pravidlá)**
 - Tá sa nachádza hore – vedľa „Data“ a „Usage“.

4. Uprav pravidlá podľa potreby

Príklad:

```
{
  "rules": {
    ".read": true,
    ".write": true
  }
}
```

Po dokončení vývoja je však veľmi dôležité tieto pravidlá sprísniť, aby nedošlo k zneužitiu alebo poškodeniu dát.

Arduino kód

Arduino kód slúži na ovládanie dvoch LED diód na základe hodnôt uložených v cloude – v službe Firebase Realtime Database. Doska Arduino UNO WiFi Rev 2 sa po štarte pripojí na Wi-Fi a následne pravidelne (každých 300 ms) kontroluje databázu, či sa zmenil stav uzlov leds/led1 a leds/led2.

Ak sa hodnota zmení, kód aktualizuje stav príslušnej LED diódy – zapne ju alebo vypne. Vďaka tomu je možné LED diódy ovládať na diaľku cez internet, napríklad pomocou webového rozhrania.

JE POTREBNÉ VYPLNIŤ ÚDAJE:

- **WIFI_SSID:** názov Vašej WiFi siete
- **WIFI_PASS:** heslo Vašej WiFi siete
- **DB_HOST:** `tvoj-projekt-default-rtdb.firebaseio.com` (bez `https://`)
- **DB_SECRET:**

"Settings -> Project settings -> Service accounts -> Database secrets -> Secrets"

```
#include <WiFiNINA.h>
#include <Firebase_Arduino_WiFiNINA.h>

/* ----- 1. Nastavenia ----- */
#define WIFI_SSID "TVOJA_SIET"
#define WIFI_PASS "TVOJE_HESLO"

#define DB_HOST "tvoj-projekt-default-rtdb.firebaseio.com"
#define DB_SECRET "Settings -> Project settings -> Service accounts -> Database secrets -> Secrets"

#define LED1_PIN 5 // LED 1
#define LED2_PIN 6 // LED 2
#define POLL_MS 300 // ms
```

```

/* ----- 2. Globálne ----- */

FirebaseData fbdo;

uint8_t led1State = 0, led2State = 0; // 0 = OFF, 1 = ON
uint32_t lastPoll = 0;
bool dbInitialized = false;

/* ----- 3. HW ----- */

void writeHW() {
    digitalWrite(LED1_PIN, led1State ? HIGH : LOW);
    digitalWrite(LED2_PIN, led2State ? HIGH : LOW);
}

/* ----- 4. Setup ----- */

void setup() {
    Serial.begin(9600);
    pinMode(LED1_PIN, OUTPUT);
    pinMode(LED2_PIN, OUTPUT);
    writeHW();

    /* Wi-Fi */
    WiFi.begin(WIFI_SSID, WIFI_PASS);
    Serial.print("Wi-Fi");
    while (WiFi.status() != WL_CONNECTED) {
        Serial.print('.');
        delay(300);
    }
    Serial.println(" OK");

    /* Firebase */
    Firebase.begin(DB_HOST, DB_SECRET, WIFI_SSID, WIFI_PASS);
    Firebase.reconnectWiFi(true);

    /* Inicializácia DB - vytvorenie ciest ak neexistujú */
    if (!dbInitialized) {
        if (Firebase.setInt(fbdo, "leds/led1", 0)) {
            Serial.println("Vytvorený leds/led1 s hodnotou 0");
        } else {
            Serial.print("Chyba pri vytváraní leds/led1: ");
            Serial.println(fbdo.errorReason());
        }
    }
}

```



```

}

if (Firebase.setInt(fbdo, "leds/led2", 0)) {
  Serial.println("Vytvorený leds/led2 s hodnotou 0");
} else {
  Serial.print("Chyba pri vytváraní leds/led2: ");
  Serial.println(fbdo.errorReason());
}

dbInitialized = true;
}

Serial.println("Pripravené – čakám na príkazy z webu.");
}

/* ----- 5. Loop ----- */
void loop() {
  if (millis() - lastPoll >= POLL_MS) {
    lastPoll = millis();

    /* LED 1 */
    if (Firebase.getInt(fbdo, "leds/led1")) {
      uint8_t v = fbdo.intData() ? 1 : 0;
      if (v != led1State) {
        led1State = v;
        writeHW();
        Serial.print("LED1 -> "); Serial.println(v ? "ON" : "OFF");
      }
    } else {
      Serial.print("Chyba LED1: "); Serial.println(fbdo.errorReason());
    }

    /* LED 2 */
    if (Firebase.getInt(fbdo, "leds/led2")) {
      uint8_t v = fbdo.intData() ? 1 : 0;
      if (v != led2State) {
        led2State = v;
        writeHW();
        Serial.print("LED2 -> "); Serial.println(v ? "ON" : "OFF");
      }
    }
  }
}

```

```

    } else {
        Serial.print("Chyba LED2: "); Serial.println(fbdo.errorReason());
    }
}
}
}

```

Server kód

Súbor index.html predstavuje jednoduché webové rozhranie, ktoré umožňuje používateľovi na diaľku ovládať dve LED diódy pomocou prepínačov (checkboxov). Po kliknutí na checkbox sa zmení hodnota v Firebase Realtime Database, ktorú následne Arduino číta a reaguje zapnutím alebo vypnutím LED.

Ako vytvoriť index.html:

1. Otvorte si **textový editor** (napr. Notepad, VS Code, Sublime Text).
2. Skopírujte doň HTML kód (nachádza sa nižšie).
3. Nahradte vo Firebase konfigurácii v časti initializeApp vlastné údaje projektu.

apiKey: "Settings -> Project settings -> Service accounts -> Database secrets -> Secrets",

databaseURL: "<https://name-of-yourproject-rtdb.firebaseio.com>"

4. Súbor uložte pod názvom: index.html
5. Dvojklikom otvorte súbor v prehliadači.

Webová stránka bude fungovať lokálne a komunikovať priamo s Firebase databázou – bez potreby servera.

```

<!doctype html>
<html lang="sk">
<meta charset="utf-8">
<title>LED ovládač (0/1)</title>

<style>
body{ font-family:sans-serif;text-align:center;margin-top:4rem}
.wrap{display:inline-block;margin:1rem}
label{display:block;font-size:1.2rem;margin-bottom:.3rem}
input[type=checkbox]{width:60px;height:30px;cursor:pointer}
</style>

<script type="module">
import { initializeApp }      from
    "https://www.gstatic.com/firebasejs/11.6.1/firebase-app.js";
import { getDatabase, ref, onValue,

```

```

        set                } from
    "https://www.gstatic.com/firebasejs/11.6.1/firebase-database.js";

/* Zmeňte na svoj projekt */
const app = initializeApp({
  apiKey:    "Settings -> Project settings -> Service accounts -> Database secrets -> Secrets",
  databaseURL: "https://name-of-yourproject-rtdb.firebaseio.com"
});
const db = getDatabase(app);

const ck1 = document.getElementById("ck1");
const ck2 = document.getElementById("ck2");

/* DB → UI */
onValue(ref(db, "leds/led1"), snap => { if (snap.exists()) ck1.checked = snap.val() === 1; });
onValue(ref(db, "leds/led2"), snap => { if (snap.exists()) ck2.checked = snap.val() === 1; });

/* UI → DB */
ck1.addEventListener("change", () => set(ref(db, "leds/led1"), ck1.checked ? 1 : 0));
ck2.addEventListener("change", () => set(ref(db, "leds/led2"), ck2.checked ? 1 : 0));
</script>

<body>
  <h1>Ovládanie LED diód</h1>

  <div class="wrap">
    <label for="ck1">LED 1</label>
    <input type="checkbox" id="ck1">
  </div>

  <div class="wrap">
    <label for="ck2">LED 2</label>
    <input type="checkbox" id="ck2">
  </div>
</body>
</html>

```

DDoS útok (Distributed Denial of Service) je typ kybernetického útoku, pri ktorom útočník zahlučuje cieľový systém – napríklad webový server, sieťové zariadenie alebo službu –

obrovským množstvom požiadaviek z viacerých zdrojov naraz. Cieľom je vyradiť službu z prevádzky, spomaliť ju alebo úplne znemožniť jej dostupnosť pre legitímnych používateľov.

Kľúčové charakteristiky DDoS útoku:

- **Distribovaný charakter:** útok je realizovaný z mnohých zariadení (často infikovaných botnetom), čím sa zvyšuje jeho účinnosť.
- **Zameranie na dostupnosť:** nejde o krádež dát, ale o narušenie funkčnosti služby.
- **Typy útokov:**
 - **Volumetrické útoky** – zahltenie siete veľkým objemom dát.
 - **Protokolové útoky** – zneužitie slabín v sieťových protokoloch (napr. TCP SYN flood).
 - **Útoky na aplikačnej vrstve** – cielené na konkrétne aplikácie (napr. HTTP požiadavky).

V kontexte IoT:

IoT zariadenia, vrátane mikrokontrolérov, sú často zraniteľné voči zneužitiu ako súčasť botnetu (napr. Mirai). Zabezpečenie týchto zariadení – silné heslá, aktualizácie firmvéru, obmedzenie prístupu – je kľúčové na prevenciu ich zneužitia pri DDoS útokoch.

Simulácia DDoS útoku pre výučbu (bezpečná a etická forma)

Cieľ: Ukázať študentom, ako môže nadmerné množstvo požiadaviek ovplyvniť dostupnosť jednoduchého webového servera (napr. ESP32 s Firebase).

Obsah simulácie:

- Skript v Pythone, ktorý opakovane posiela požiadavky na Firebase databázu (napr. prepína LED).
- Meranie odozvy servera a prípadné zahltenie.
- Diskusia o tom, prečo je to problém a ako sa tomu dá predísť.

Tento dokument bol vypracovaný pre project Nitrianske kompetenčné centrum pre kybernetickú bezpečnosť (17R05-04-V01-00003)



Nitrianske kompetenčné centrum kybernetickej bezpečnosti



Financované
Európskou úniou
NextGenerationEU

PLÁN [OBNOVY]