Sprawozdanie zprojektu

Systemy wbudowane

SW 2026: System kontroli dostępu RFID z logowaniem i uprawnieniami

14.01.2026

Prowadzący: mgr inż. Norbert Łukaniszyn

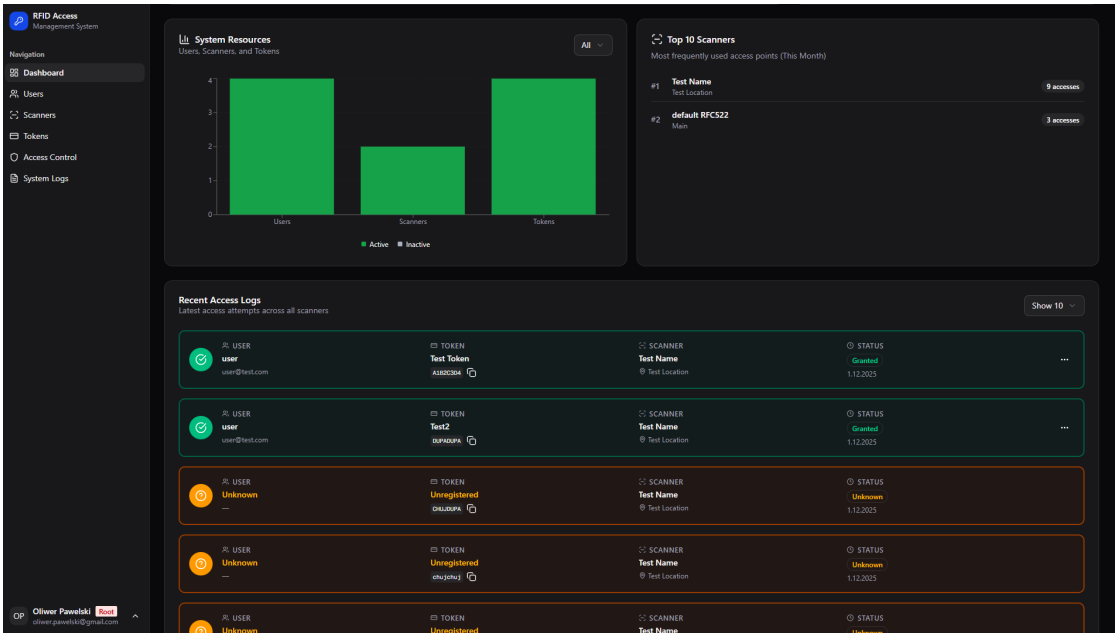
Grupa: 34INF-SSI SP/B

Naraziński Dawid
112102@stud.uz.zgora.pl
Pawelski Oliwer
112109@stud.uz.zgora.pl

Spis treści

1 Wstęp

Głównym założeniem niniejszego projektu jest zaprojektowanie oraz budowa zaawansowanego systemu kontroli dostępu, który wykorzystuje technologię identyfikacji radiowej (RFID) do autoryzacji użytkowników. System umożliwia sterowanie fizycznym ryglem drzwi za pomocą kart oraz breloków przypisanych do konkretnych osób zarejestrowanych w systemie. Rozwiązanie oparto na nowoczesnym mikrokontrolerze XIAO ESP32C3, który dzięki zintegrowanemu modułowi Wi-Fi pozwala na komunikację z zewnętrzną bazą danych. Kluczowym aspektem projektu jest rozróżnianie poziomów uprawnień dla różnych grup użytkowników, takich jak pracownicy, kierownicy czy administratorzy. System oferuje pełną administrację zdalną poprzez interfejs sieciowy, co pozwala na zarządzanie dostępami bez konieczności fizycznej ingerencji w urządzenie. Logowanie zdarzeń odbywa się w czasie rzeczywistym, co umożliwia bieżący monitoring prób wejścia do zabezpieczonych pomieszczeń. Projekt posiada również walor dydaktyczny, pozwalając na zgłębienie zagadnień związanych z protokołami komunikacyjnymi (SPI), obsługą baz danych PostgreSQL oraz programowaniem nowoczesnych frameworków webowych. Cel projektu został zdefiniowany jako dostarczenie bezpiecznego, stabilnego i skalowalnego systemu, który może znaleźć zastosowanie w warunkach rzeczywistych, np. w biurach czy laboratoriach. Finalny produkt integruje warstwę sprzętową z chmurowym backendem, zapewniając pełną synchronizację danych.



Rysunek 1: Interfejs Dashboard - główny panel z podsumowaniem zasobów systemowych i statystykami dostępu

2 Zagadnienia teoretyczne

Technologia RFID (Radio-Frequency Identification) wykorzystuje fale radiowe do przesyłania danych między czytnikiem a tagiem (kartą/brelokiem). W projekcie zastosowano czytnik RC522, który pracuje na częstotliwości 13.56 MHz i komunikuje się z mikrokontrolerem za pomocą magistrali SPI.

Rysunek 2: Czytnik RFID-RC522 zamontowany na drewnianej szafce

Mikrokontroler XIAO ESP32C3, będący sercem systemu, oparty jest na architekturze RISC-V i oferuje wsparcie dla Wi-Fi oraz Bluetooth, co jest kluczowe dla funkcji IoT.

2.1 Inicjalizacja systemu

Funkcja `setup()` wykonywana jest jednorazowo przy starcie mikrokontrolera i odpowiada za konfigurację wszystkich używanych peryferiów oraz nawiązanie połączenia z siecią Wi-Fi.

```
1 void setup() {
2     Serial.begin(9600);
3     delay(1000);
4 
5     Serial.println("\n=== RFID Access Control - HTTPS ===\n");
6 
7     pinMode(SOLENOID_PIN, OUTPUT);
8     digitalWrite(SOLENOID_PIN, LOW);
```

```
9
10      SPI.begin(8, 9, 10, RFID_SS_PIN);
11      mfrc522.PCD_Init();
12
13      Serial.print("Laczenie z WiFi...");
14      WiFi.begin(ssid, password);
15
16      int attempts = 0;
17      while (WiFi.status() != WL_CONNECTED && attempts < 30) {
18          delay(500);
19          Serial.print(".");
20          attempts++;
21      }
22
23      if (WiFi.status() == WL_CONNECTED) {
24          Serial.println(" OK");
25          Serial.print("IP: ");
26          Serial.println(WiFi.localIP());
27      } else {
28          Serial.println(" BLAD!");
29      }
30
31      client.setInsecure();
32
33      Serial.println("\nSystem gotowy - przyloz karte RFID\n");
34  }
```

Listing 1: Inicjalizacja systemu - konfiguracja peryferiów połączenie Wi-Fi oraz przygotowanie czytnika RFID

2.2 Odczyt kart RFID

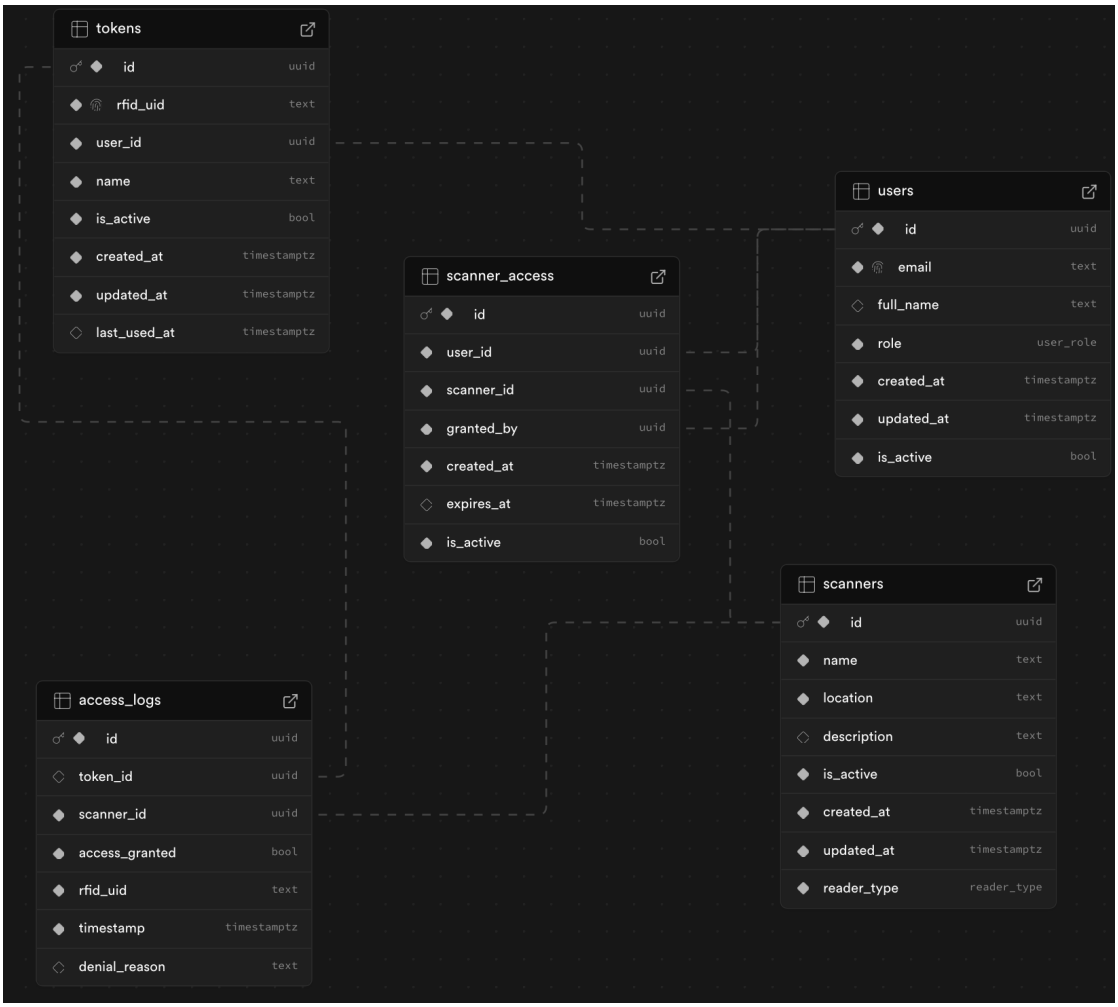
Główna pętla programu nieustannie monitoruje obecność nowych kart RFID w zasięgu czytnika. Po wykryciu karty następuje odczyt unikalnego identyfikatora (UID) i weryfikacja uprawnień dostępu poprzez wywołanie API.

```
1   void loop() {
2       if (!mfrc522.PICC_IsNewCardPresent()) {
3           delay(50);
4           return;
5       }
6
7       if (!mfrc522.PICC_ReadCardSerial()) {
8           delay(50);
9           return;
10      }
11
12      getCardID(currentCardID);
13      mfrc522.PICC_HaltA();
14      mfrc522.PCD_StopCrypto1();
```

```
    delay(100);

    if (strcmp(currentCardID, lastCardID) == 0 &&
        (millis() - lastCardTime) < 2000) {
        return;
    }

    strcpy(lastCardID, currentCardID);
    lastCardTime = millis();

    Serial.print("\n[KARTA] Token: ");
    Serial.println(currentCardID);

    if (checkAccess(currentCardID)) {
        Serial.println("[DOSTEP] Przyznany!");
        openDoor();
    } else {
        Serial.println("[DOSTEP] Odmowiony!");
    }

    mfrc522.PCD_Init();
}
```

Listing 2: Główna pętla programu - wykrywanie i odczyt kart RFID z mechanizmem anty-powtórzeń

Komunikacja sieciowa odbywa się poprzez protokół HTTPS, przesyłając żądania w formacie JSON do API serwera. Backend systemu został zrealizowany w oparciu o Supabase, który dostarcza bazę danych PostgreSQL oraz mechanizmy autentykacji i kontroli ról (RBAC). Architektura bazy danych obejmuje relacyjne powiązania między użytkownikami, ich tokenami oraz uprawnieniami do konkretnych skanerów.



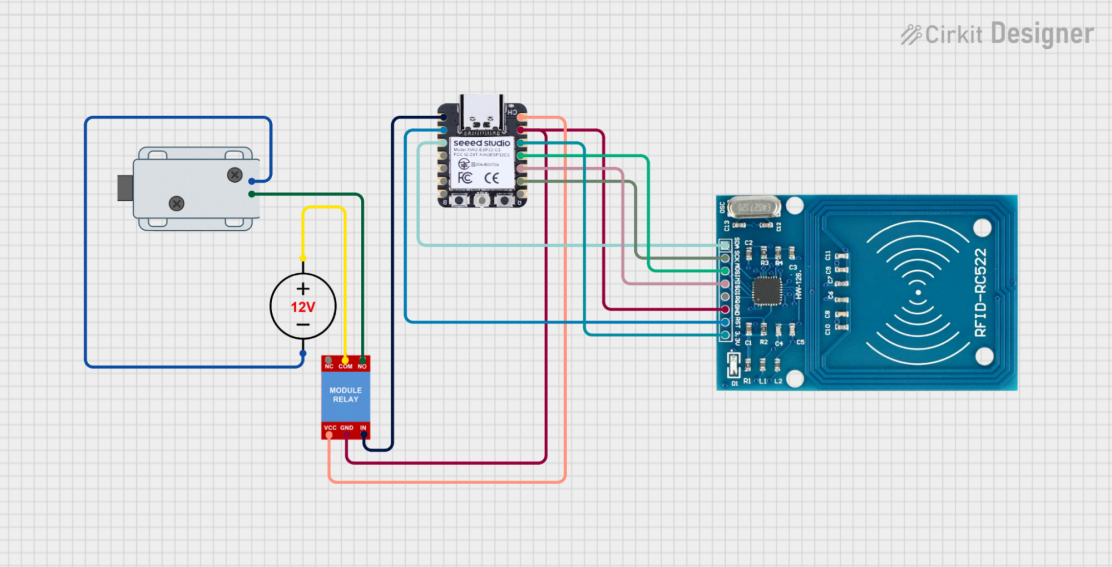
Rysunek 3: Diagram struktury bazy danych PostgreSQL z relacjami między tabelami (tokens, users, scanners, scanner_access, access_logs)

Interfejs użytkownika zbudowano w frameworku Next.js, co zapewnia szybkie renderowanie stron i responsywność. System ról obejmuje poziomy: Root, Admin oraz User, gdzie każda ranga posiada ściśle określone kompetencje edycyjne. Bezpieczeństwo fizyczne zapewnia elektrozamek (solenoid) 12V sterowany poprzez tranzystor pełniący rolę klucza elektronicznego. System uwzględnia również czasowe ograniczenia dostępu, co pozwala na nadawanie uprawnień z datą wygaśnięcia. Logowanie zdarzeń obejmuje nie tylko udane próby wejścia, ale także odmowy wraz z podaniem przyczyny (np. brak uprawnień, nieznany tag). Całość wdrożona jest na platformie Vercel, co gwarantuje wysoką dostępność panelu administracyjnego. System został zaprojektowany z myślą o skalowalności, co umożliwia obsługę wielu skanerów w ramach jednej infrastruktury sieciowej.

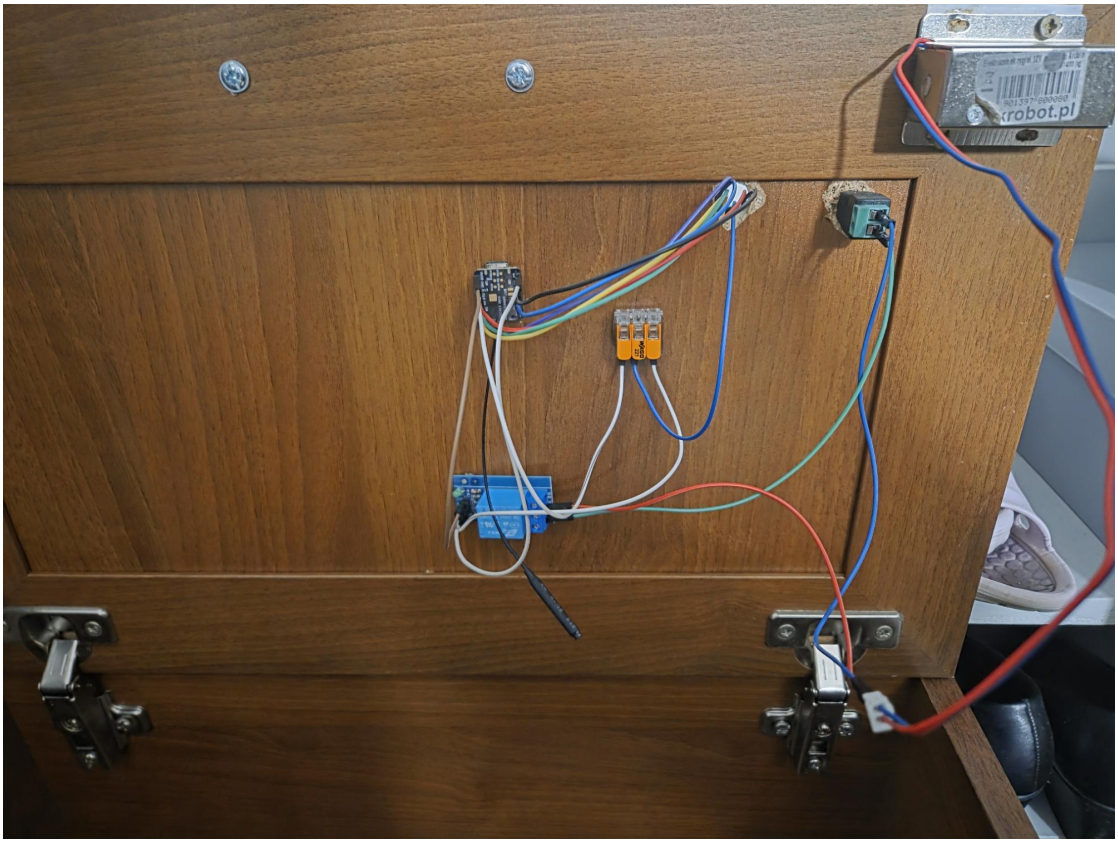
3 Implementacja i usprawnienia

Warstwa sprzętowa została zmontowana na podstawie schematu połączeń wykorzystującego piny GPIO mikrokontrolera ESP32C3 dla magistrali SPI (MISO, MOSI, SCK,

SDA) oraz sygnału sterującego przekaźnikiem (GPIO2).



Rysunek 4: Schemat połączeń elektrycznych systemu w Cirkit Designer (ESP32C3, RC522, przekaźnik, elektrozamek)



Rysunek 5: Wnętrze szafki z zamontowanymi komponentami systemu - ESP32C3, moduł przekaźnika, elektrozamek i okablowanie

3.1 Komunikacja HTTPS z API

Mikrokontroler komunikuje się z backendem poprzez zabezpieczone połączenie HTTPS. Każde przyłożenie karty powoduje wysłanie żądania POST do endpointu `/api/v1/access` z danymi skanera i tokenu RFID.

```
bool checkAccess(char* cardID) {
    if (WiFi.status() != WL_CONNECTED) {
        Serial.println("BLAD: WiFi rozlaczone!");
        return false;
    }

    Serial.print("Sprawdzanie dostepu...");

    if (!client.connect(server, httpsPort)) {
        Serial.println(" BLAD polaczenia!");
        return false;
    }

    String jsonBody = "{\"scanner\":\"";
    jsonBody += scannerId;
    jsonBody += "\",\"token\":\"";
    jsonBody += cardID;
    jsonBody += "\"}";

    client.print("POST /api/v1/access HTTP/1.1\r\n");
    client.print("Host: ");
    client.print(server);
    client.print("\r\n");
    client.print("Content-Type: application/json\r\n");
    client.print("Content-Length: ");
    client.print(jsonBody.length());
    client.print("\r\n");
    client.print("Connection: close\r\n\r\n");
    client.print(jsonBody);

    unsigned long timeout = millis();
    while (!client.available()) {
        if (millis() - timeout > 10000) {
            Serial.println(" Timeout!");
            client.stop();
            return false;
        }
        delay(10);
    }

    bool inBody = false;
    bool granted = false;

    while (client.available()) {
        String line = client.readStringUntil('\n');
```

```
46
47          if (line.length() <= 1) {
48              inBody = true;
49              continue;
50          }
51
52          if (inBody && line.indexOf("\"granted\":true") > -1) {
53              granted = true;
54          }
55      }
56
57      client.stop();
58      Serial.println(granted ? " OK" : " ODMOWA");
59      return granted;
60  }
```

Listing 3: Weryfikacja dostępu - komunikacja HTTPS z API i parsowanie odpowiedzi JSON

3.2 Sterowanie elektrozamkiem

Po pozytywnej weryfikacji uprawnień aktywowany jest elektromagnes (solenoid), który odryglowuje zamek na czas 3 sekund, umożliwiając otwarcie drzwi.

```
1  void openDoor() {
2      Serial.println("Otwieranie drzwi...");
3      digitalWrite(SOLENOID_PIN, HIGH);
4      delay(3000);
5      digitalWrite(SOLENOID_PIN, LOW);
6      Serial.println("Zamknieto\n");
7  }
```

Listing 4: Sterowanie elektrozamkiem - aktywacja przekaźnika i opóźnienie czasowe

- **Szyfrowanie HTTPS:** Zabezpieczenie komunikacji między ESP32 a API za pomocą certyfikatów SSL/TLS, co uniemożliwia podsłuchanie danych tokenów w sieci lokalnej.
- **Mechanizm anty-powtórzeń:** System zapobiega wielokrotnemu odczytowi tej samej karty w krótkim odstępie czasu (2 sekundy), eliminując przypadkowe wielokrotne otwarcia.
- **Timeout połączenia:** Implementacja limitu czasowego (10 sekund) dla żądań HTTP zabezpiecza przed zablokowaniem urządzenia przy problemach z siecią.

3.3 Panel zarządzania webowego

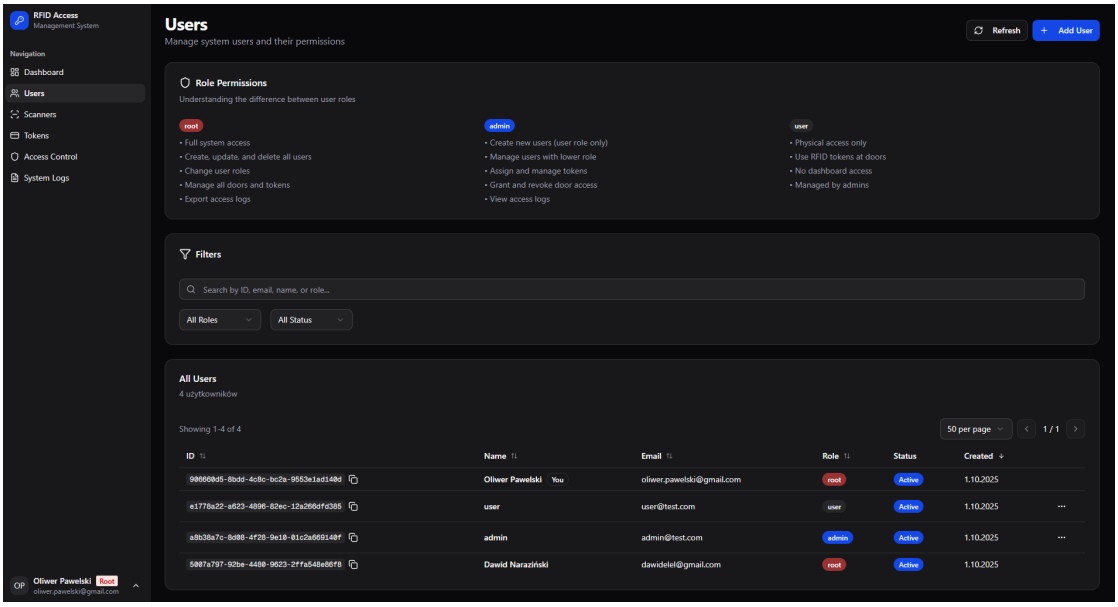
System posiada rozbudowany webowy panel administracyjny z kontrolą dostępu opartą na rolach użytkowników (RBAC - Role-Based Access Control). Interfejs umożliwia kompleksowe zarządzanie wszystkimi aspektami systemu kontroli dostępu przez

przeglądarkę internetową. Strona publiczna `/login` służy do logowania do systemu za pomocą email i hasła. Dostęp do panelu administracyjnego wymaga posiadania roli **root** lub **admin**, przy czym zakres uprawnień różni się w zależności od poziomu dostępu.

Ścieżka	Moduł	Root	Admin
`/dashboard`	Strona główna	Statystyki, wykresy, logi, akcje	
`/dashboard/users`	Użytkownicy	CRUD wszystkie role, reset haseł	CRUD tylko rola 'user', reset haseł
`/dashboard/scanners`	Skanery RFID	CRUD, konfiguracja lokalizacji	Tylko odczyt
`/dashboard/tokens`	Tokeny RFID	CRUD, przypisywanie użytkownikom	CR, aktywacja/dezaktywacja
`/dashboard/access`	Kontrola dostępu	Zarządzanie uprawnieniami, datami wygaśnięcia	
`/dashboard/logs`	Logi	Przeglądanie, filtrowanie, eksport CSV	Przeglądanie, filtrowanie

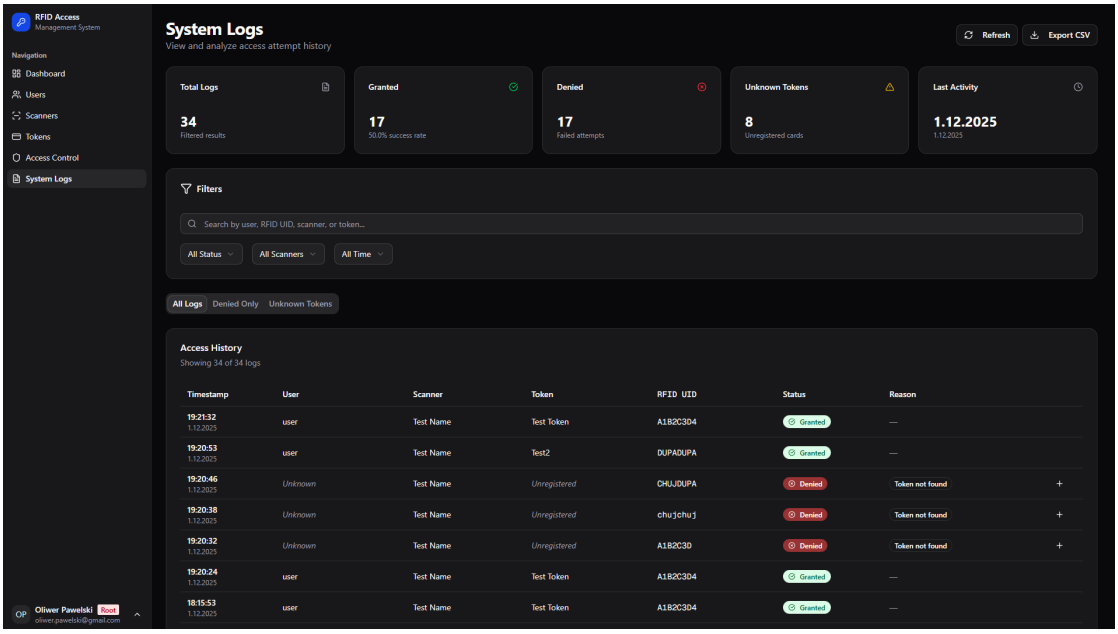
Tabela 1: Macierz uprawnień panelu administracyjnego (CRUD: Create/Read/Update-/Delete).

Oprogramowanie mikrokontrolera realizuje algorytm odczytu UID, wysłania go do endpointu `/api/v1/access` i oczekiwania na odpowiedź typu boolean `access.granted`. Panel webowy umożliwia eksport logów do formatu CSV oraz podgląd statystyk w formie wykresów. System wykorzystuje biblioteki: `MFRC522` do obsługi czytnika RFID, `WiFiClientSecure` do komunikacji HTTPS oraz `SPI` do interfejsu komunikacyjnego z czytnikiem.



Rysunek 6: Interfejs Dashboard - wizualizacja zasobów systemowych i statystyk dostępu w formie wykresów

4 Testy i wnioski



Rysunek 7: Interfejs System Logs - strona logów dostępu z podziałem na udane, odrzucone i nieznane tokeny

Testy systemu uprawnień potwierdziły poprawne blokowanie dostępu dla użytkowników z wygasłą datą ważności tokena oraz dla osób nieprzypisanych do danego skanera. Interfejs administracyjny poprawnie wyświetla logi w czasie rzeczywistym, a filtrowanie zdarzeń działa płynnie. Wnioski z realizacji wskazują, że połączenie systemów wbudowanych z technologiami cloud (Supabase) znacząco upraszcza zarządzanie rozproszoną infrastrukturą kontroli dostępu. Projekt jest gotowy do wdrożenia w małej skali, a potencjalna rozbudowa mogłaby obejmować integrację z systemami alarmowymi oraz aplikację mobilną.

5 Kosztorys

Poniższa tabela przedstawia szacunkowe koszty komponentów użytych do budowy jednego punktu dostępowego.

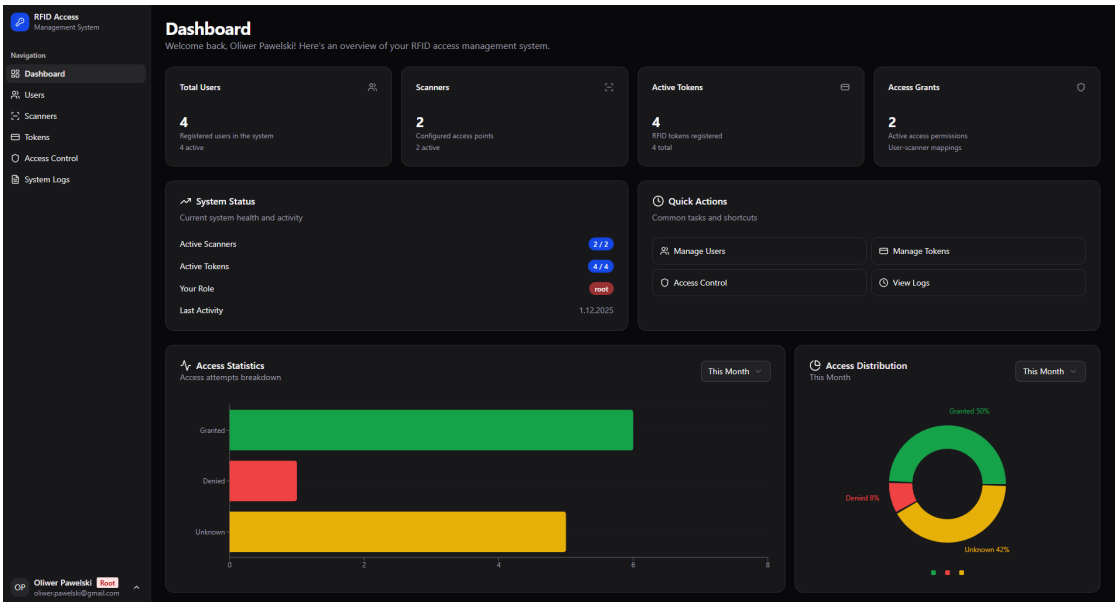
6 Załącznik

W niniejszym załączniku przedstawiono dodatkową dokumentację fotograficzną projektu, która szczegółowo ilustruje interfejsy webowe systemu zarządzania dostępem.

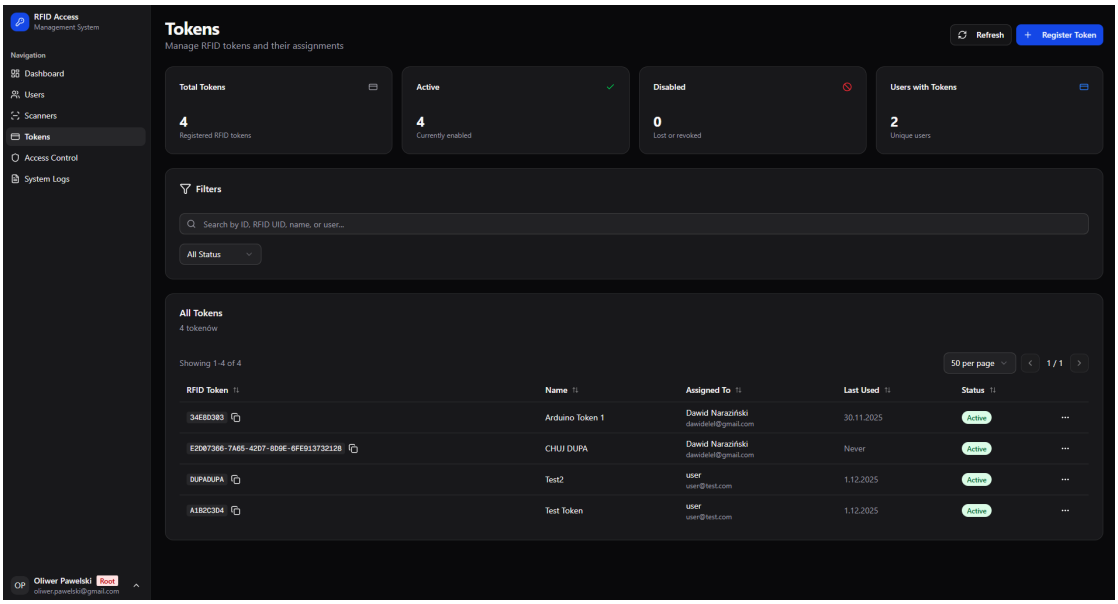
Element	Ilość	Cena jedn.	Suma
Seeed Studio XIAO ESP32C3	1 szt.	35,00 zł	35,00 zł
Czytnik RFID-RC522 + karta/brelok	1 szt.	15,00 zł	15,00 zł
Elektrozamek (Solenoid) 12V	1 szt.	25,00 zł	25,00 zł
Moduł przekaźnika 5V / Tranzystor	1 szt.	8,00 zł	8,00 zł
Zasilacz 12V DC / Przetwornica	1 szt.	20,00 zł	20,00 zł
Obudowa, przewody, drobne elementy	-	20,00 zł	20,00 zł
Suma całkowita			**123,00 zł**

Tabela 2: Szacunkowy kosztorys sprzętowy projektu.

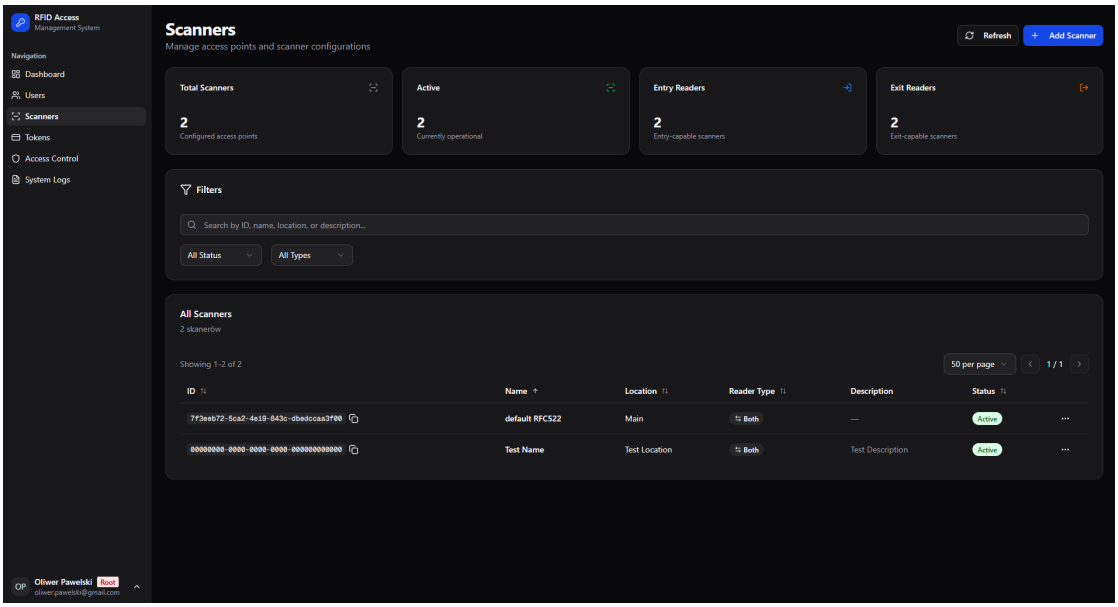
6.1 Interfejsy webowe systemu



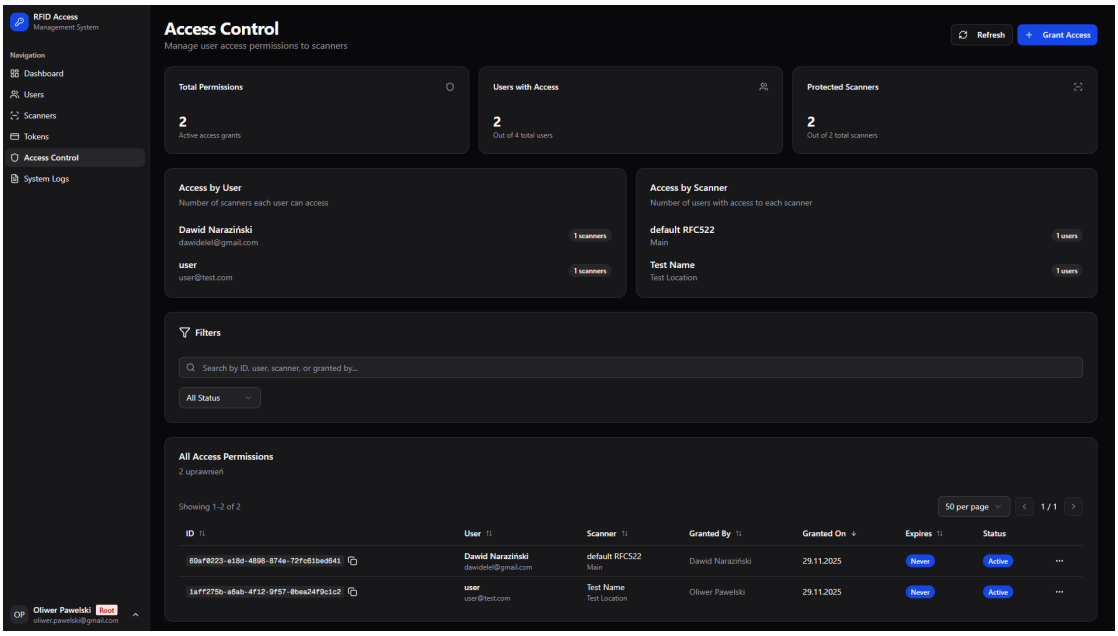
Rysunek 8: Interfejs Users - zarządzanie użytkownikami systemu i ich rolami (root, admin, user)



Rysunek 9: Interfejs Tokens - zarządzanie zarejestrowanymi tokenami RFID i ich przypisaniami



Rysunek 10: Interfejs Scanners - konfiguracja punktów dostępu i skanerów RFID



Rysunek 11: Interfejs Access Control - zarządzanie uprawnieniami użytkowników do poszczególnych skanerów

6.2 Kod źródłowy projektu

Pełny kod źródłowy projektu, w tym oprogramowanie mikrokontrolera oraz interfejs webowy, dostępny jest w repozytorium GitHub: https://github.com/Guliveer/RFID-access-manager