



WYDZIAŁ ELEKTRONIKI I TECHNIK INFORMACYJNYCH

PROJEKTOWANIE APLIKACJI INTERNETOWYCH I MOBILNYCH

---

# Portal do nauki mechanizmów bezpieczeństwa z użyciem programu nmap i snort

---

Jan Gajownik  
Tomasz Skok  
Mikita Zhydko  
Cezary Witkowski

1 czerwca 2025

# Spis treści

<b>1</b>	<b>Temat projektu</b>	<b>2</b>
<b>2</b>	<b>Opis ogólnej koncepcji rozwiązania</b>	<b>2</b>
<b>3</b>	<b>Ogólną architekturę systemu</b>	<b>2</b>
<b>4</b>	<b>Listę wymaganych i zrealizowanych funkcji z krótkim wyjaśnieniem sposobu realizacji</b>	<b>3</b>
<b>5</b>	<b>Opis aplikacji webowej</b>	<b>5</b>
5.1	Architektura aplikacji . . . . .	5
5.2	Struktura projektu . . . . .	6
5.3	Widoki i powiązane z nimi funkcje aplikacji . . . . .	6
<b>6</b>	<b>Opis aplikacji back-end</b>	<b>7</b>
6.1	Architektura aplikacji . . . . .	7
6.2	Struktura projektu . . . . .	8
6.3	Modele danych . . . . .	8
6.4	Endpointy API . . . . .	9
6.5	Zabezpieczenia (dostęp do endpointów, przechowywanie haseł) . . . . .	9
<b>7</b>	<b>Opis aplikacji mobilnej (UAIM)</b>	<b>9</b>
7.1	Architektura aplikacji . . . . .	9
7.2	Lista wykorzystywanych bibliotek zewnętrznych . . . . .	9
7.3	Projekt widoków . . . . .	9
7.4	Dokumentacja klas . . . . .	9
<b>8</b>	<b>Opis pliku konfiguracyjnego Docker</b>	<b>9</b>
<b>9</b>	<b>Wyniki działania aplikacji</b>	<b>10</b>
9.1	Webowej . . . . .	10
9.2	Mobilnej (UAIM) . . . . .	17
<b>10</b>	<b>Podsumowanie</b>	<b>18</b>

# 1 Temat projektu

Portal do nauki mechanizmów bezpieczeństwa (programy nmap i snort oraz dwie maszyny wirtualne - skanująca i skanowana)

- Logowanie do portalu
- Rejestracja
- Przekazywanie komend konfigurujących i uruchamiających skanowanie programem nmap (na wzór nakładki graficznej Zenmap)
- Przekazywanie komend konfigurujących i uruchamianie/zatrzymywanie różnych usług (email, ftp, ssh, www) oraz uruchamianie mechanizmu IDS/IPS programu snort
- Wyświetlanie wyników skanowania na ekranie, ściąganie plików logów

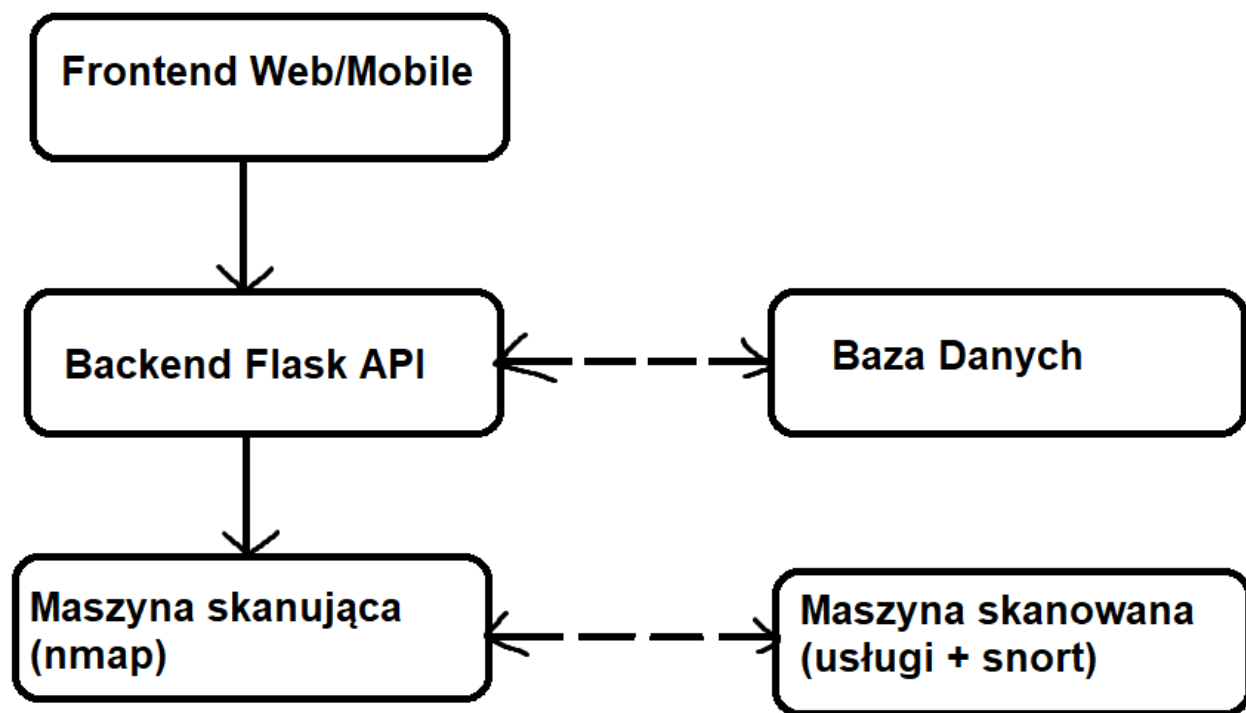
# 2 Opis ogólnej koncepcji rozwiązania

Projekt zakłada stworzenie portalu edukacyjnego, który umożliwi użytkownikom praktyczną naukę mechanizmów bezpieczeństwa sieciowego z wykorzystaniem narzędzi takich jak nmap (do skanowania sieci) oraz snort (system IDS/IPS). Całość działa w środowisku złożonym z dwóch maszyn wirtualnych: skanującej (atakującej) oraz skanowanej (ofiary/serwerem).

# 3 Ogólną architekturę systemu

Ogólna architektura systemu opiera się na modelu klient-serwer z podziałem na kilka głównych komponentów:

- Frontend (klient)  
Aplikacja webowa (React) – umożliwia użytkownikom interakcję z portalem przez przeglądarkę (logowanie, rejestracja, uruchamianie skanów, przeglądanie wyników, pobieranie logów). Aplikacja mobilna (np. Android) – opcjonalnie, pozwala na dostęp do wybranych funkcji portalu z poziomu urządzenia mobilnego.
- Backend (serwer aplikacji)  
Flask (Python) – realizuje logikę biznesową, obsługuje żądania HTTP od klientów, zarządza użytkownikami, autoryzacją (JWT), przekazuje polecenia do maszyn wirtualnych, udostępnia wyniki i pliki.
- Baza danych (np. SQLite) – przechowuje dane użytkowników, historię skanów, logi itp.
- Środowisko laboratoryjne (maszyny wirtualne)  
Maszyna skanująca (atakująca) – uruchamia polecenia nmap, wysyła wyniki do backendu. Maszyna skanowana (serwer) – posiada konfigurowalne usługi (FTP, SSH, WWW, SMTP) oraz system IDS/IPS (snort), który można uruchamiać/zatrzymywać z poziomu portalu.
- Komunikacja  
Frontend komunikuje się z backendem przez REST API (HTTP/HTTPS, JSON). Backend komunikuje się z maszynami wirtualnymi poprzez polecenia systemowe (np. przez Docker lub SSH).



Rysunek 1: Schemat blokowy

#### 4 Listę wymaganych i zrealizowanych funkcji z krótkim wyjaśnieniem sposobu realizacji

- **Logowanie do portalu**

**Wymagania:** Portal posiada system uwierzytelniania użytkowników. Użytkownik loguje się za pomocą e-mailu i hasła, a po poprawnej autoryzacji otrzymuje token JWT, który pozwala na korzystanie z chronionych zasobów portalu.

**Realizacja:** Formularz logowania wysyła dane do `/api/login`. Po poprawnej autoryzacji backend zwraca token JWT, który jest przechowywany po stronie klienta i dołączany do kolejnych żądań.

- **Rejestracja**

**Wymagania:** Nowi użytkownicy mogą założyć konto poprzez formularz rejestracyjny. Dane są zapisywane w bazie danych, a po rejestracji użytkownik może się zalogować i korzystać z funkcjonalności portalu.

**Realizacja:** Formularz rejestracyjny na froncie przesyła dane do endpointu `/api/register` w backendzie Flask, gdzie dane są zapisywane w bazie danych.

- **Przekazywanie komend konfiguracyjnych i uruchamiających skanowanie programem nmap**

**Wymagania:** Portal umożliwia użytkownikowi wybór różnych typów skanowania (np. TCP, UDP, stealth) oraz dodatkowych opcji (np. detekcja systemu operacyjnego). Po zatwierdzeniu, odpowiednie polecenie nmap jest przekazywane do maszyny skanującej, gdzie jest wykonywane. Wyniki skanowania są odbierane przez backend i prezentowane użytkownikowi w czytelnej formie (na wzór Zenmap).

**Realizacja:** Interfejs graficzny (wzorem Zenmap) pozwala wybrać typ skanu. Wybrane opcje są przesyłane do backendu, który uruchamia odpowiednie polecenie nmap na maszynie skanującej (np. przez Docker exec). Do tego użytkownik wybiera usługi w interfejsie, a backend przesyła odpowiednie komendy do maszyny wirtualnej (np. service ssh start/stop).

- **Przekazywanie komend konfiguracyjnych i uruchamianie/zatrzymywanie różnych usług oraz uruchamianie mechanizmu IDS/IPS programu snort**

**Wymagania:** Użytkownik może z poziomu portalu włączać/wyłączać wybrane usługi na maszynie skanowanej (np. serwer WWW, FTP, SSH, SMTP). Dodatkowo może uruchomić lub zatrzymać system IDS/IPS (snort), który monitoruje ruch sieciowy i wykrywa potencjalne ataki. Wszystkie te akcje są realizowane poprzez przekazywanie odpowiednich komend do maszyny wirtualnej.

**Realizacja:** Przycisk w interfejsie wysyła żądanie do backendu, który uruchamia lub zatrzymuje snorta odpowiednią komendą systemową.

- **Wyświetlanie wyników skanowania na ekranie, ściąganie plików logów oraz wyświetlanie historii skanowania**

**Wymagania:** Wyniki skanowania nmap są prezentowane użytkownikowi na stronie portalu w formie tabeli lub tekstu. Użytkownik ma również możliwość pobrania plików logów (np. wyników skanowania w formacie XML lub logów snorta) na swój komputer.

**Realizacja:** Backend przetwarza wynik XML z nmap i zwraca go w formacie tekstowym lub tabelarycznym do frontendu, gdzie jest wyświetlany użytkownikowi. Może on go zapisać i wynik takiego skanu zostanie zapisany w bazie danych i będzie dostępny do przeglądania po zalogowaniu w zakładce Account. Użytkownik ma również możliwość pobrania pliku przez kliknięcie przycisku, który wywołuje endpoint backendu zwracający plik do pobrania.

## 5 Opis aplikacji webowej

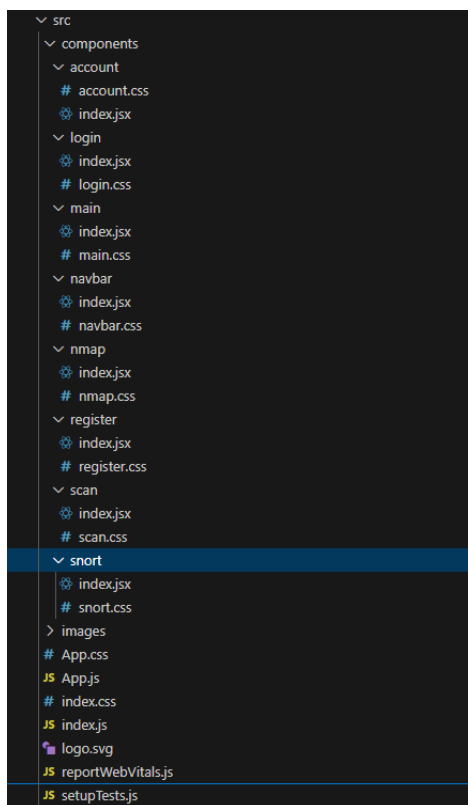
Aplikacja webowa frontendowa została zrealizowana w technologii React.js i stanowi graficzny interfejs użytkownika portalu do nauki mechanizmów bezpieczeństwa sieci komputerowych. Jej główne cechy i funkcjonalności to:

- Interfejs użytkownika
- Obsługa logowania i rejestracji
- Konfiguracja i zarządzanie usługami
- Uruchamianie skanowania nmap
- Zarządzanie systemem IDS/IPS (Snort)
- Prezentacja i obsługa wyników
- Komunikacja z backendem

### 5.1 Architektura aplikacji

Architektura aplikacji webowej frontendowej opiera się na technologii React.js i została zaprojektowana w sposób komponentowy. Wszystkie funkcje, takie jak logowanie, rejestracja, konfiguracja usług, uruchamianie skanowania nmap, zarządzanie Snortem czy prezentacja wyników, realizowana zostały napisane jako osobne komponenty React. Komunikacja z backendem odbywa się poprzez wywołania REST API za pomocą fetch, a autoryzacja użytkownika realizowana jest przez token JWT przechowywany w localStorage i dołączany do nagłówek żądań. Stan aplikacji (np. aktywna zakładka, wyniki skanowania, konfiguracja usług) zarządzany jest lokalnie w komponentach za pomocą hooków useState i useEffect. Interfejs użytkownika jest responsywny i podzielony na sekcje tematyczne (usługi, nmap, snort), a wyniki skanowania oraz logi prezentowane są w czytelnej formie z możliwością pobrania lub zapisania. Całość zapewnia płynną obsługę użytkownika bez przeładowywania strony, a logika prezentacji i obsługi zdarzeń jest oddzielona od logiki biznesowej, która znajduje się po stronie backendu.

## 5.2 Struktura projektu



Rysunek 2: Struktura front-endu

## 5.3 Widoki i powiązane z nimi funkcje aplikacji

- **Widok logowania i rejestracji**

Umożliwia użytkownikowi zalogowanie się lub założenie nowego konta. Po poprawnym zalogowaniu użytkownik uzyskuje dostęp do reszty funkcji portalu.

- **Widok główny (panel skanowania)**

Podzielony na zakładki: Services, Nmap, Snort. Każda zakładka odpowiada za inną funkcjonalność.

- **Zakładka "Services"**

Umożliwia włączanie i wyłączanie usług (SSH, HTTP, SMTP, FTP, RPC, DNS) na maszynie wirtualnej. Użytkownik widzi listę usług i może je uruchamiać/zatrzymywać przyciskiem.

- **Zakładka "Nmap"**

Pozwala wybrać typ skanowania (TCP, Stealth, UDP) oraz opcję detekcji systemu operacyjnego. Użytkownik uruchamia skanowanie przyciskiem "Start Scanning". Po zakończeniu skanowania wyświetlane są wyniki oraz dostępne są przyciski do pobrania surowego wyniku (Download Raw XML) lub zapisania skanu (Save Scan).

- **Zakładka "Snort"**

Umożliwia włączenie lub wyłączenie systemu IDS/IPS Snort na maszynie skanowanej. Użytkownik może aktywować ochronę sieci jednym kliknięciem.

- **Widok wyników skanowania**

Wyniki skanowania prezentowane są w czytelnej formie tekstowej. Użytkownik może pobrać wyniki lub zapisać je do historii.

- **Widok historii skanów (jeśli takowa istnieje)**

Pozwala przeglądać zapisane wcześniej wyniki skanowania i pobierać pliki.

- **Widok strony głównej**

Przedstawienie tematyki strony

- **Widok strony narzędzi (tools)**

Opis narzędzi nmap i snort

## 6 Opis aplikacji back-end

Aplikacja backendowa została zaprojektowana jako REST API do zarządzania użytkownikami, konfiguracją usług systemowych, kontrolą Snorta, wykonywaniem skanów Nmap oraz przechowywaniem i pobieraniem wyników tych skanów. System działa w środowisku kontenerowym Docker, integrując maszyny wirtualne symulujące scenariusz atakującego i obrońcy (ATTACKER\_NAME i DEFENDER\_NAME).

### 6.1 Architektura aplikacji

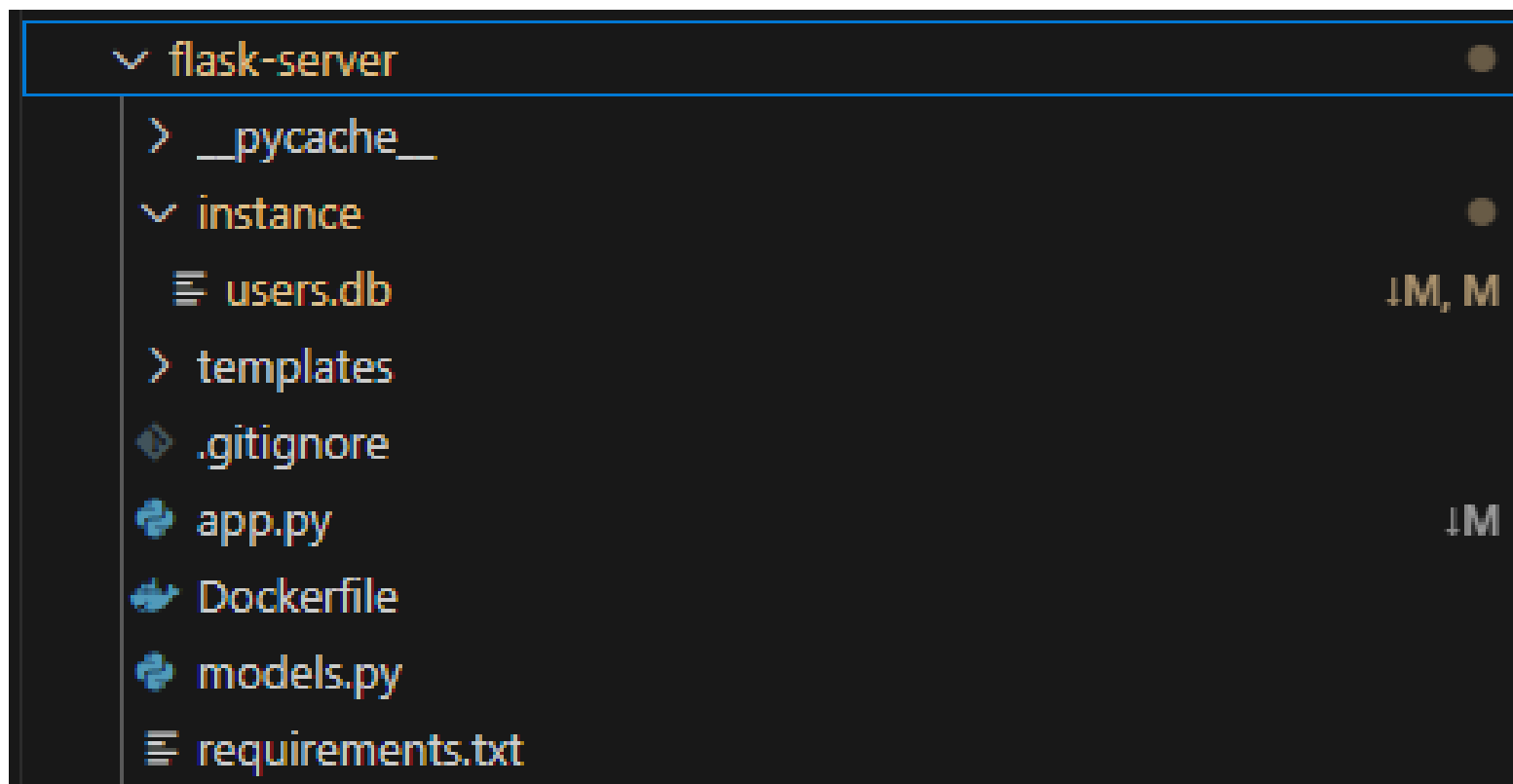
Aplikacja składa się z dwóch głównych komponentów uruchamianych w kontenerach Docker:

- **flask-server** – serwer backendowy napisany we Flasku, pełniący funkcję API.
- **linux\_machine\_1** – kontener symulujący maszynę atakującą (Kali Linux), z którego wykonywane są ataki (nmap).
- **linux\_machine\_2** – kontener symulujący maszynę broniącą, na której uruchamiane są usługi oraz system IDS/IPS Snort.

Komunikacja pomiędzy kontenerami odbywa się przez Docker bridge network. Backend zarządza usługami w kontenerach za pomocą komend wykonywanych przez `docker exec`.



## 6.2 Struktura projektu



Rysunek 3: Struktura back-endu

## 6.3 Modele danych

W projekcie wykorzystywane są dwa modele danych, zdefiniowane w SQLAlchemy:

- User:
  - id – unikalny identyfikator
  - username – nazwa użytkownika (adres e-mail)
  - password – hasło (przechowywane jako tekst, patrz: zabezpieczenia)
- ScanFile:
  - id – identyfikator pliku
  - user\_id – powiązanie z użytkownikiem
  - filename, filetype, filedata, hashtag, created\_at

## 6.4 Endpointy API

Poniżej przedstawiono najważniejsze dostępne endpointy API:

- POST /api/register – rejestracja użytkownika
- POST /api/login – logowanie i uzyskanie tokenu JWT
- POST /api/services – start/stop usług systemowych (np. SSH, FTP) na maszynie broniącej
- POST /api/snort – uruchomienie lub zatrzymanie Snorta
- POST /api/scan – skan nmap (opcjonalnie z detekcją systemu operacyjnego)
- GET /api/user/scans – pobranie historii skanów danego użytkownika
- POST /api/save\_scan – zapis wyniku skanu do bazy
- GET /api/download/<scan\_id> – pobranie wyniku skanu jako plik XML

## 6.5 Zabezpieczenia (dostęp do endpointów, przechowywanie haseł)

- Dostęp do większości funkcji związanych z plikami skanów wymaga uwierzytelnienia za pomocą tokenu JWT (dekorator @jwt\_required()).
- Interakcje z kontenerami Docker odbywają się tylko z poziomu backendu, co ogranicza ryzyko nieautoryzowanego dostępu.
- Dane skanów są przechowywane w bazie i można je pobrać jedynie po autoryzacji.

## 7 Opis aplikacji mobilnej (UAIM)

### 7.1 Architektura aplikacji

### 7.2 Lista wykorzystywanych bibliotek zewnętrznych

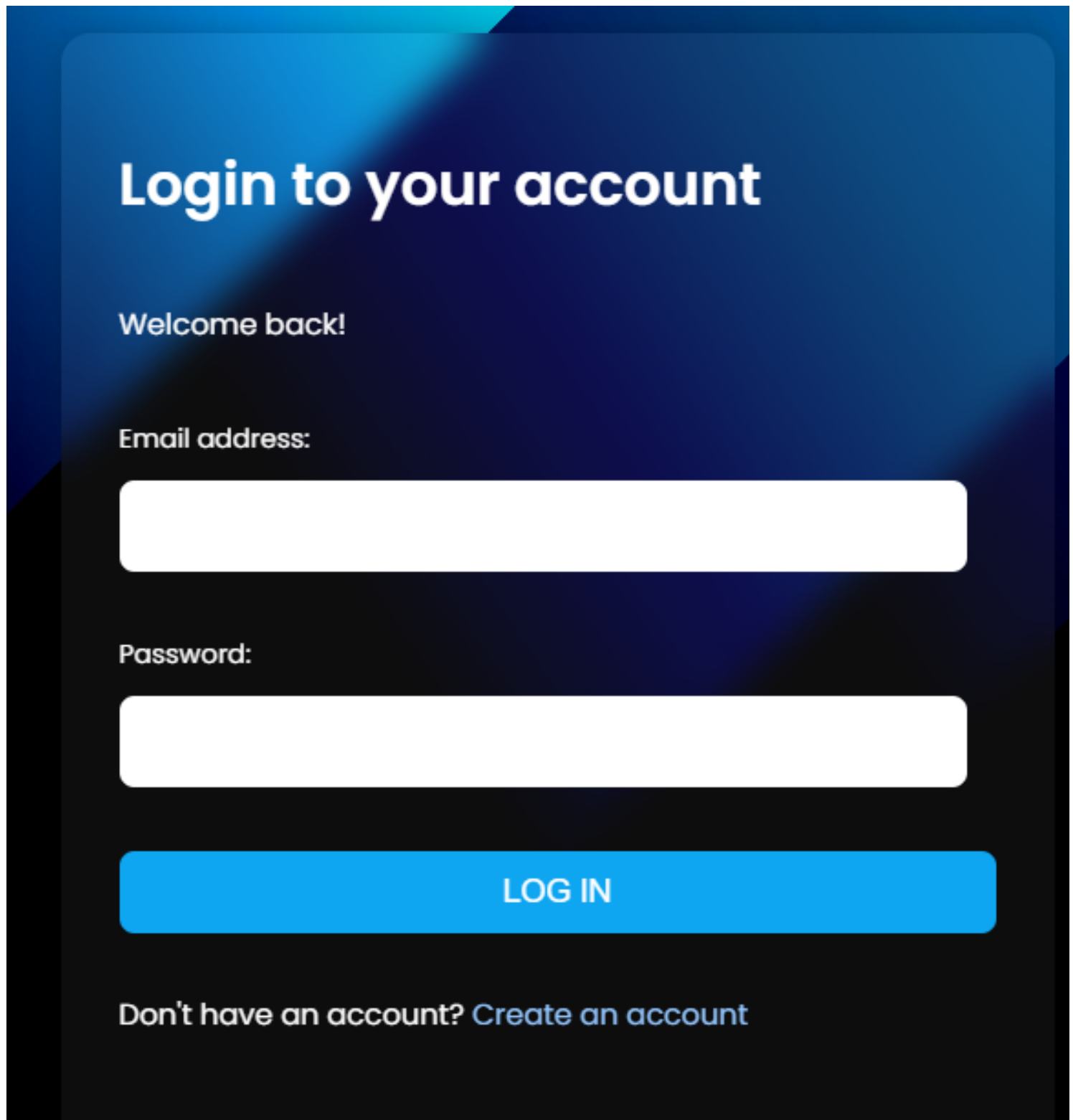
### 7.3 Projekt widoków

### 7.4 Dokumentacja klas

## 8 Opis pliku konfiguracyjnego Docker

## 9 Wyniki działania aplikacji

### 9.1 Webowej



**Login to your account**

Welcome back!

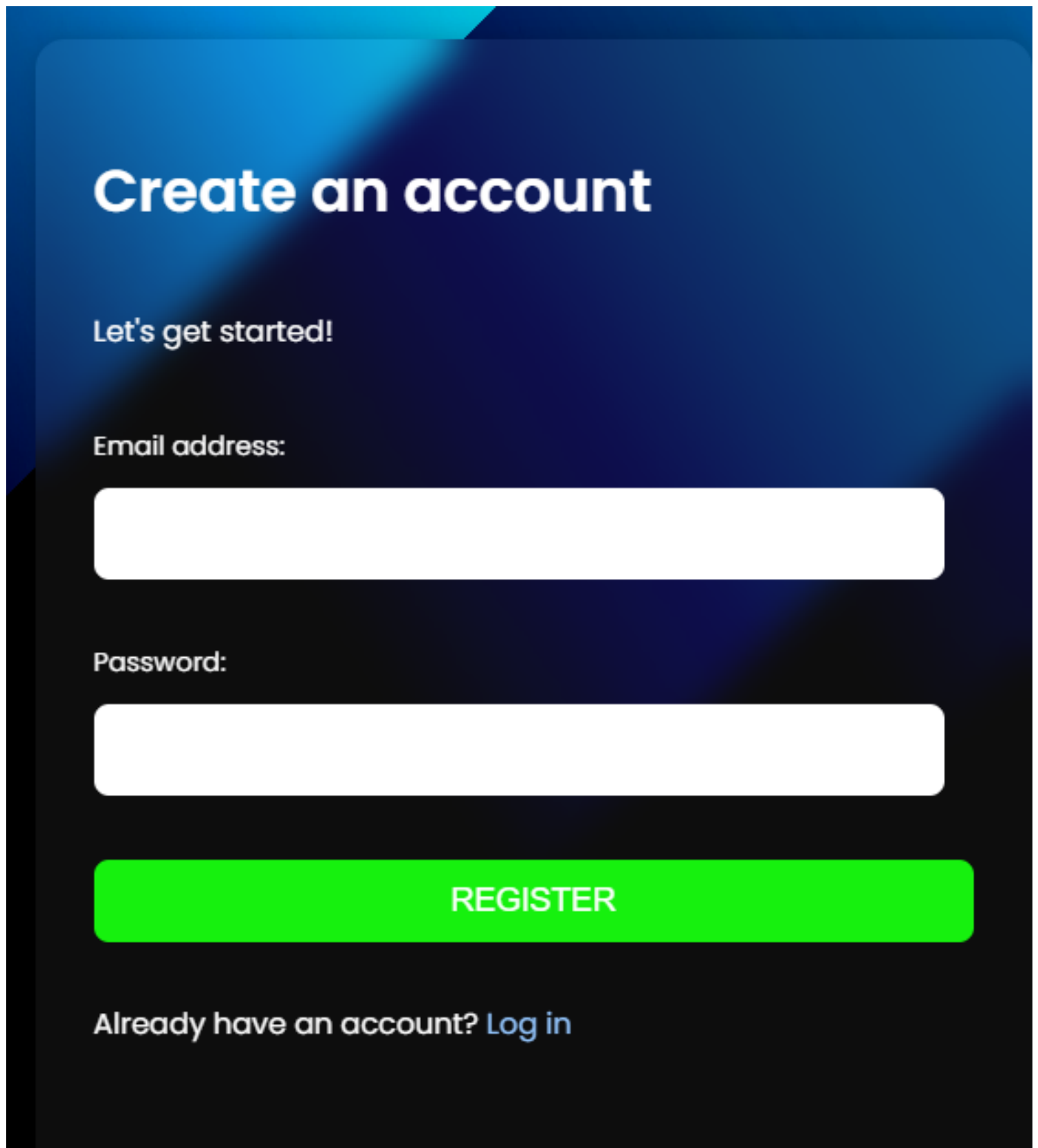
Email address:

Password:

**LOG IN**

Don't have an account? [Create an account](#)

Rysunek 4: Strona logowania



**Create an account**

Let's get started!

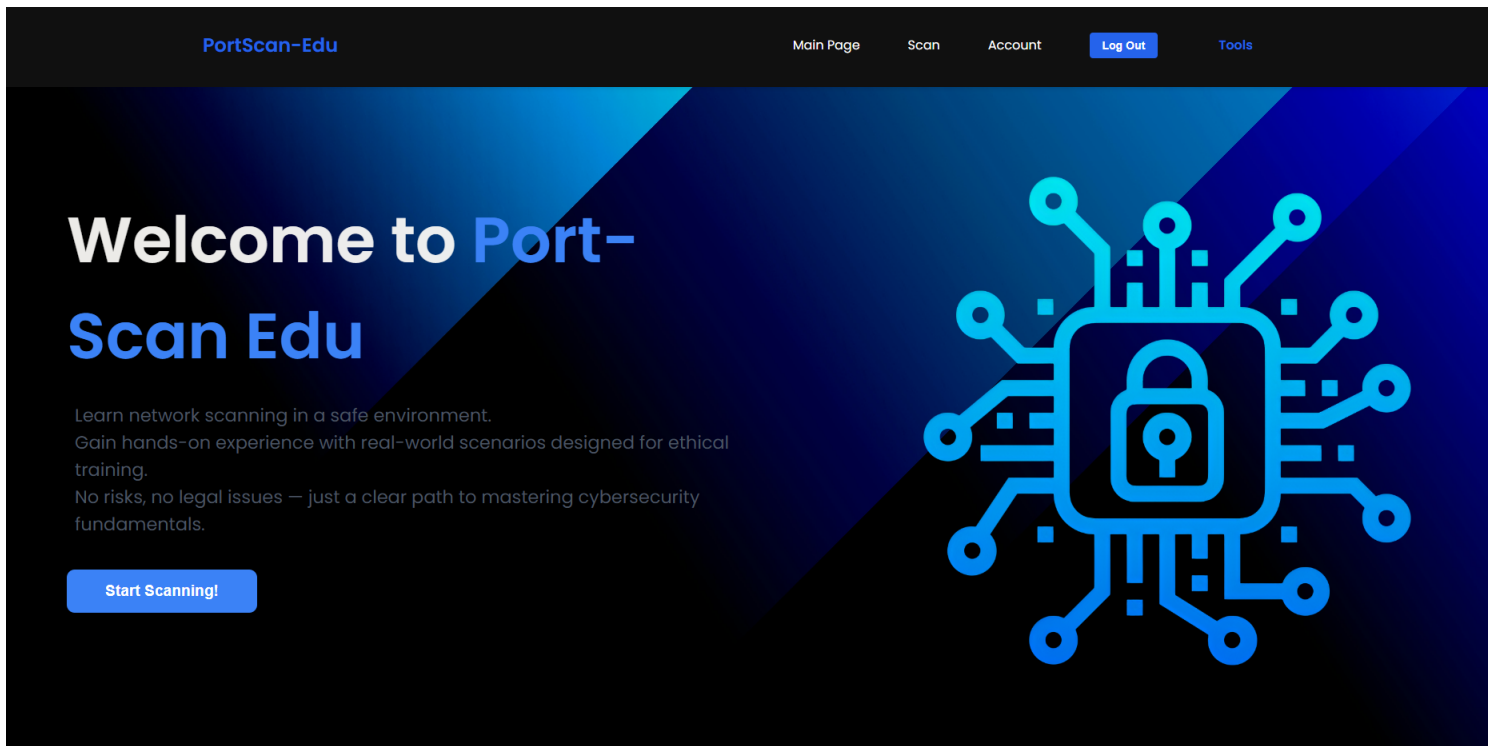
Email address:

Password:

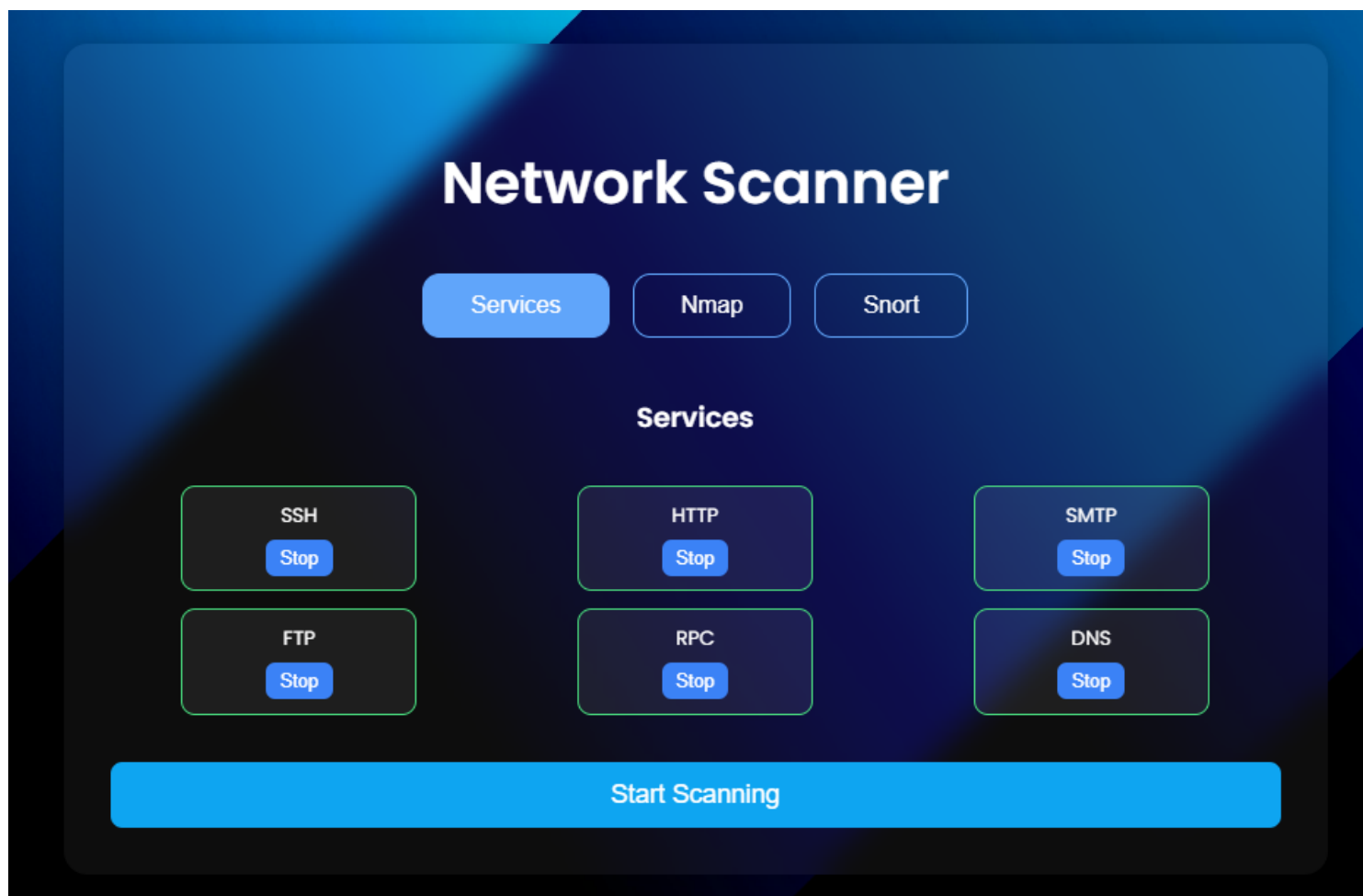
**REGISTER**

Already have an account? [Log in](#)

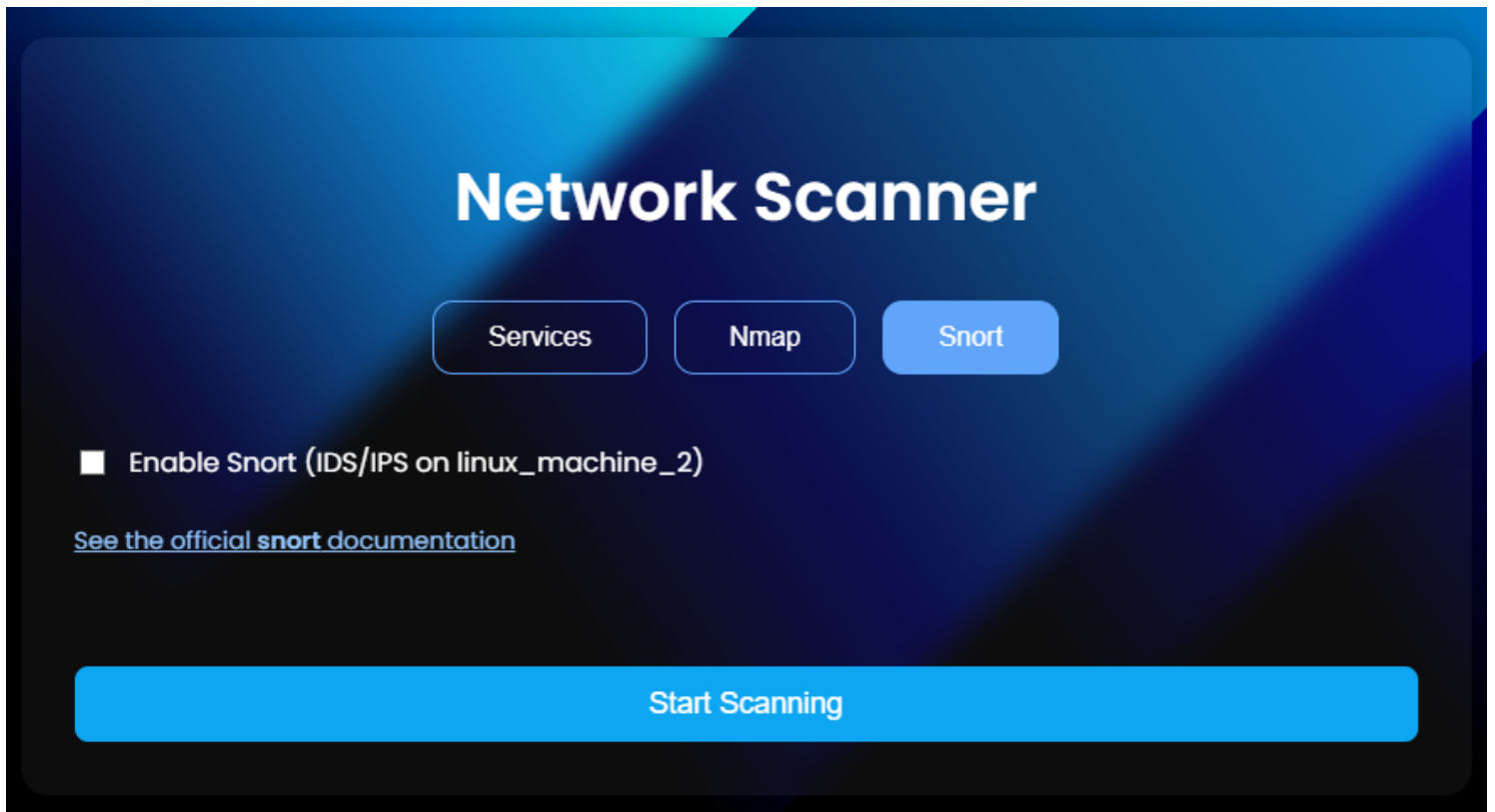
Rysunek 5: Strona rejestracji



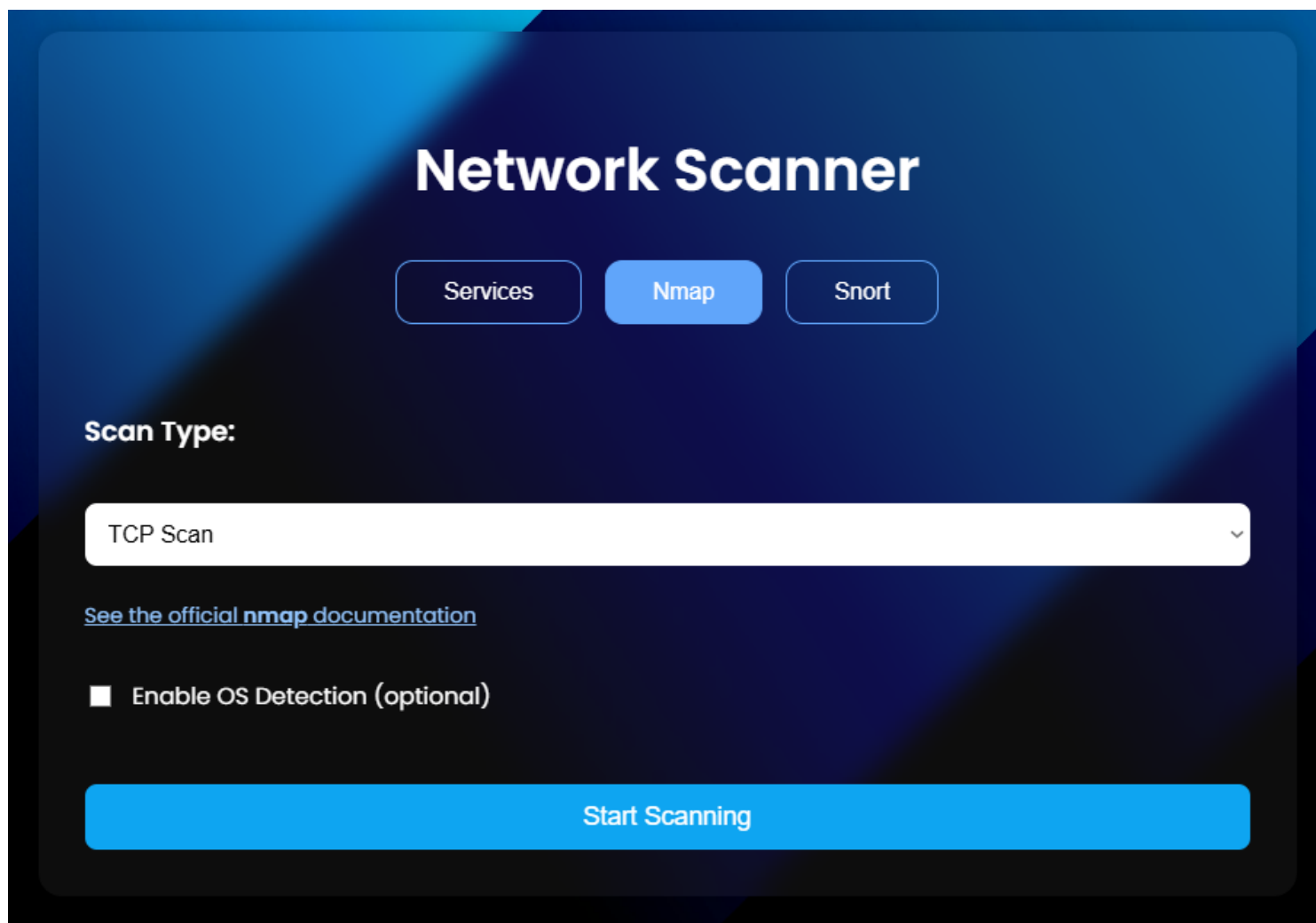
Rysunek 6: Strona główna



Rysunek 7: Wybór portów do skanowania

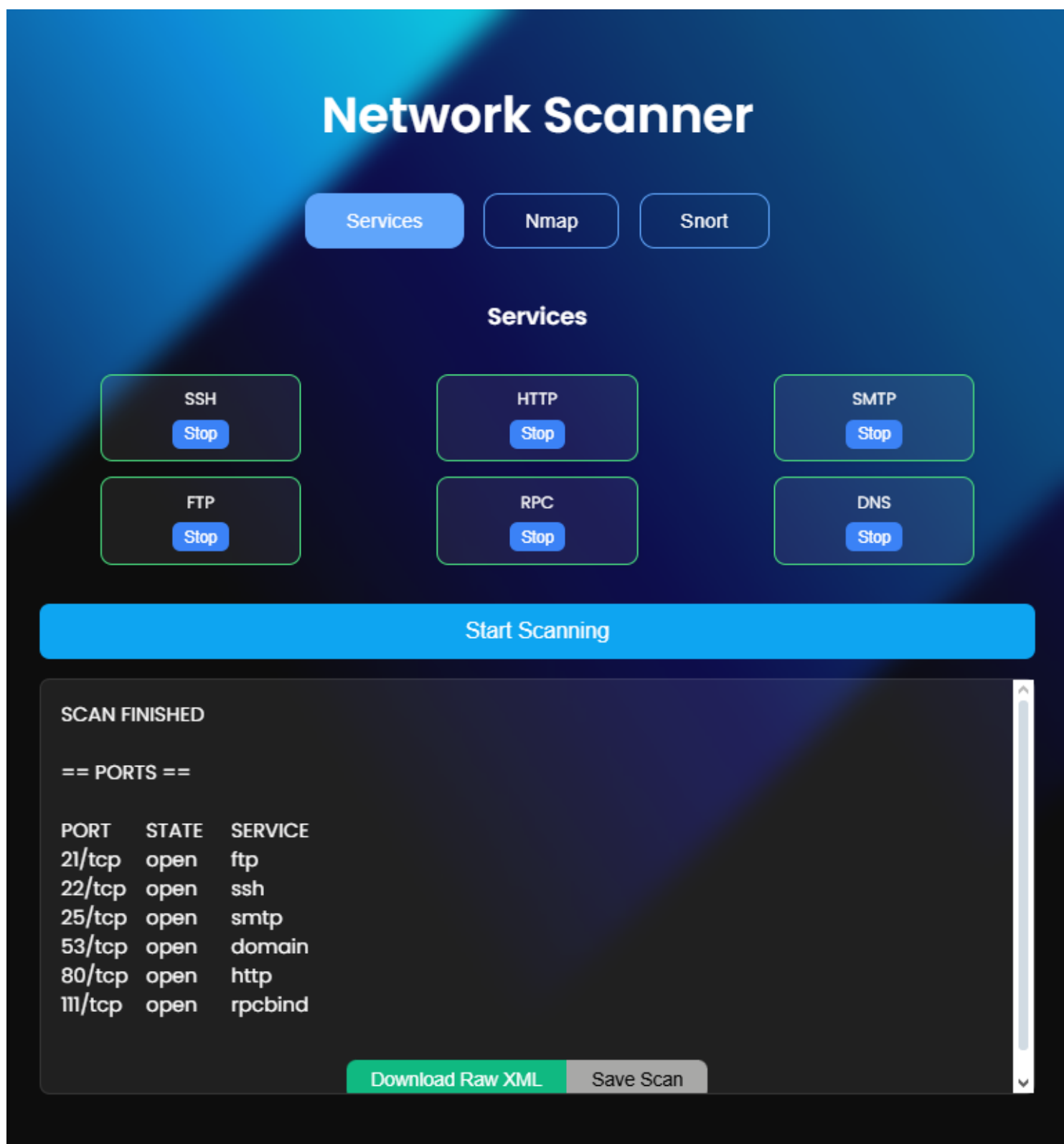


Rysunek 8: Konfiguracja snorta

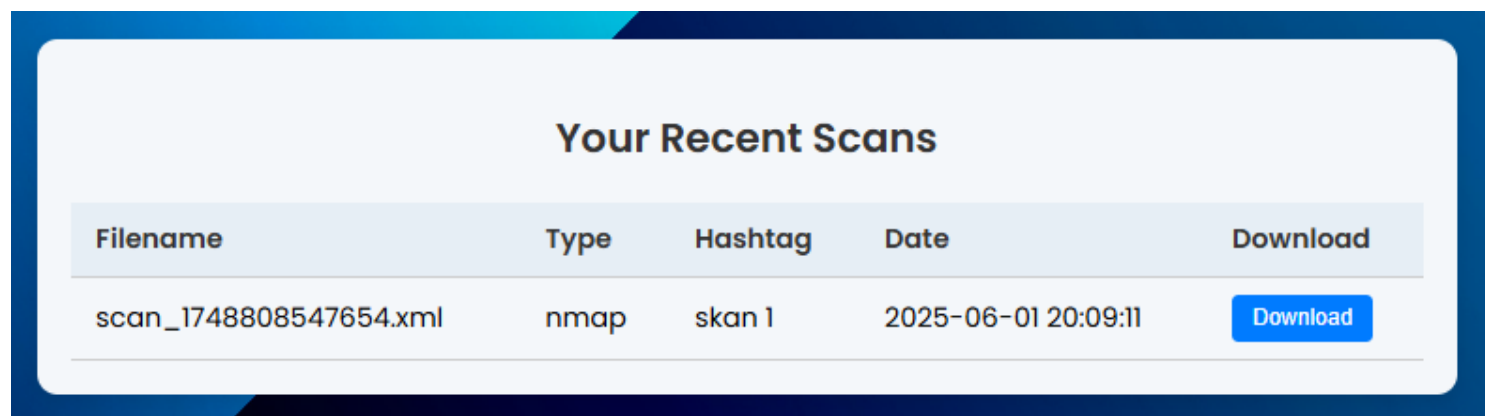


Rysunek 9: Konfiguracja nmapa





Rysunek 10: Ukończony skan



The screenshot shows a web interface titled "Your Recent Scans". It contains a table with the following columns: Filename, Type, Hashtag, Date, and Download. A single row of data is visible, representing a scan file named "scan\_1748808547654.xml" of type "nmap" with hashtag "skan 1", dated "2025-06-01 20:09:11". A blue "Download" button is located to the right of the date.

Filename	Type	Hashtag	Date	Download
scan_1748808547654.xml	nmap	skan 1	2025-06-01 20:09:11	<a href="#">Download</a>

Rysunek 11: Lista skanów - zakładka account

## 9.2 Mobilnej (UAIM)

## 10 Podsumowanie