LarvixON

LarvixON AI – diagnostyka toksyczności osocza z wykorzystaniem analizy behawioralnej larw

Autorzy: Mikołaj Kubś · Krzysztof Kulka · Martyna Łopianiak · Patryk Łuszczek

Opiekun: dr inż. Natalia Piórkowska

Streszczenie

Projekt LarvixON ma na celu zaprojektowanie i implementację innowacyjnego narzędzia diagnostycznego, które wykorzystuje sztuczną inteligencję do identyfikacji ksenobiotyków w osoczu na podstawie analizy wzorców ruchowych larw Galleria mellonella. System integruje cztery kluczowe moduły: serwer backendowy oparty na Django, aplikację kliencką we frameworku Flutter, symulator środowiska w silniku Unity oraz model hybrydowy łączący sieci CNN i LSTM.

Głównym rezultatem prac jest w pełni funkcjonalna platforma umożliwiająca użytkownikom przesyłanie nagrań wideo i otrzymywanie automatycznej klasyfikacji substancji podanej larwom. Zastosowanie symulacji pozwoliło na przetestowanie architektury modelu ML przed pozyskaniem materiału biologicznego. Opracowane rozwiązanie ma istotne znaczenie dla rozwoju toksykologii, oferując szybką i skalowalną alternatywę dla kosztownych metod laboratoryjnych. Projekt potwierdza skuteczność łączenia bioanalizy z algorytmami głębokiego uczenia w detekcji substancji psychoaktywnych.

1 WSTĘP

Diagnostyka toksykologiczna i wykrywanie ksenobiotyków w osoczu są ważne przy podejrzeniu zatrucia, przedawkowania leków lub kontaktu z nieznaną substancją. Klasyczne badania laboratoryjne są dokładne, ale wymagają specjalistycznego sprzętu i czasu, więc w sytuacjach nagłych często nie pomagają. Stąd potrzeba prostych i szybkich metod, które pozwolą lekarzowi wstępnie ocenić, czy i jakie substancje toksyczne znajdują się w organizmie pacjenta.

Projekt LarvixON proponuje wykorzystanie larw *Galleria mellonella* jako modelu biologicznego do takiej wstępnej oceny. Ten organizm jest powszechnie używany w toksykologii, ponieważ reaguje na toksyny i patogeny wyraźnymi zmianami zachowania, głównie w ruchu. Analiza tych zmian może wskazywać, jakie substancje bioaktywne znajdują się w badanej próbce.

Celem projektu jest stworzenie zautomatyzowanego systemu opartego na sztucznej inteligencji, który będzie klasyfikował próbki osocza na podstawie wzorców ruchowych larw. Zakładany czas całej analizy to maksymalnie 20 minut, przy zachowaniu wysokiej czułości i specyficzności. System ma wspierać podejmowanie decyzji klinicznych, łącząc badania nad reakcjami larw z budową kompletnej aplikacji do analizy.

Oczekiwane efekty to szybsza diagnoza, lepsze rokowania pacjentów w stanach nagłych oraz możliwość użycia tej metody jako szybkiego testu przesiewowego w szpitalach i ambulatoriach.

2 PRACE ZWIĄZANE Z TEMATEM

Dotychczasowe metody wykrywania ksenobiotyków opierają się głównie na analizie chemicznej, która mimo dużej dokładności wymaga długich procedur i specjalistycznego sprzętu. Coraz większe zainteresowanie budzą więc podejścia wykorzystujące modele biologiczne i algorytmy sztucznej inteligencji do pośredniego wykrywania substancji toksycznych. Larwy *Galleria mellonella* zyskują na popularności w toksykologii dzięki niskim kosztom, szybkim reakcjom i pewnym podobieństwom ich odpowiedzi immunologicznej do reakcji ssaków. Zmiany w ich zachowaniu, zwłaszcza w aktywności ruchowej, można łatwo mierzyć przy użyciu systemów wizyjnych, co stwarza dobre warunki do automatyzacji całego procesu diagnostycznego.

2.1 Analiza literatury i badania wstępne

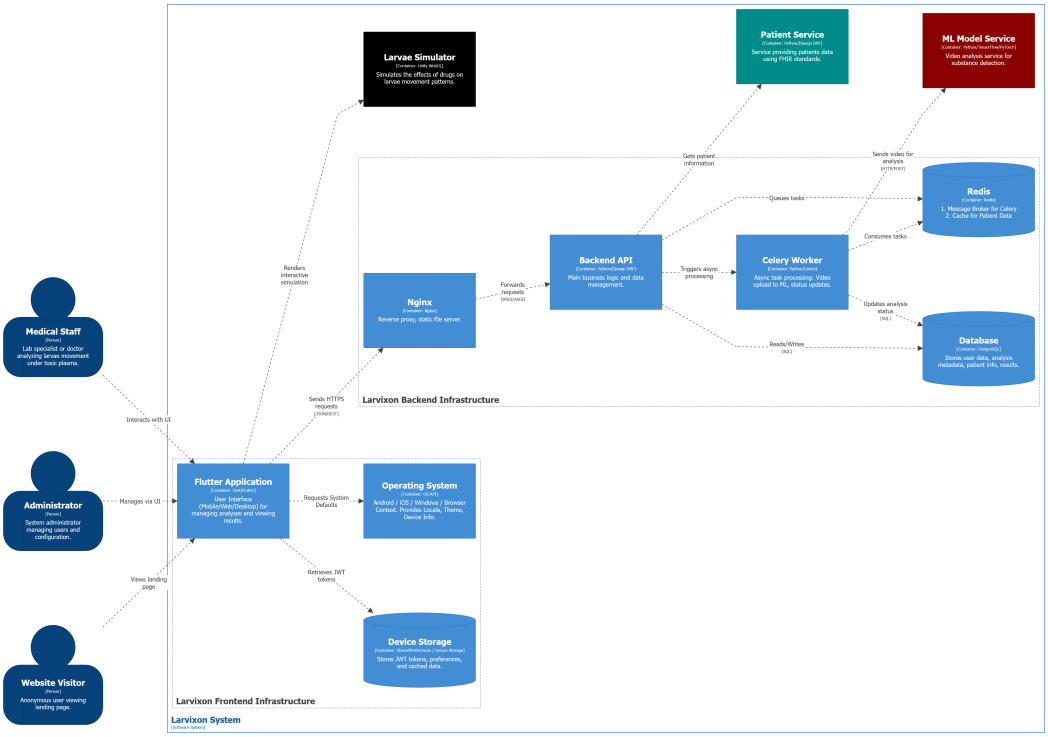
Kluczowym etapem inicjującym prace nad projektem było przeprowadzenie szczegółowej analizy literatury i podobnych, istniejących rozwiązań w obszarze diagnostyki toksykologicznej wspomaganej sztuczną inteligencją. Proces badawczy koncentrował się na trzech głównych obszarach:

1. **Modelowanie behawioralne larw *Galleria mellonella*:** Analizowano dotychczasowe wykorzystanie larw jako modelu biologicznego (*in vivo*) do oceny toksyczności. Szczególną uwagę zwrócono na publikacje opisujące **zależność zmian motorycznych od stężenia ksenobiotyków**, co stanowiło empiryczną podstawę dla naszego podejścia diagnostycznego.
2. **Nowoczesne metody komputerowego przetwarzania obrazu (*Computer Vision*):** Przeprowadzono *research* w zakresie zaawansowanych algorytmów śledzenia ruchu oraz ekstrakcji cech behawioralnych z sekwencji wideo. Analizowano efektywność tradycyjnych algorytmów śledzenia oraz nowoczesnych metod opartych na głębokim uczeniu (np. *DeepLabCut*), mając na uwadze ich wydajność i szybkość wymaganą przez system czasu rzeczywistego.
3. **Architektury modeli klasyfikacyjnych:** Zbadano nowoczesne podejścia do modelowania danych czasowo-przestrzennych. Analiza ta potwierdziła, że **hybrydowe architektury łączące konwolucyjne sieci neuronowe (CNN)** do ekstrakcji cech przestrzennych z **sieciami rekurencyjnymi (LSTM)** do modelowania dynamiki czasowej są optymalne dla problemu klasyfikacji sekwencji behawioralnych.

Wyniki analizy wykazały, że choć literatura opisuje poszczególne elementy składowe projektu (wykorzystanie *G. mellonella*, śledzenie owadów, modelowanie sekwencyjne), **brak jest kompleksowego, wdrożonego rozwiązania komercyjnego lub badawczego**, które integrowałoby wszystkie te aspekty w jedną, skalowalną platformę do szybkiej diagnostyki klinicznej. Ta luka technologiczna stanowiła główne uzasadnienie dla podjęcia prac nad projektem LarvixON.

3 WYNIKI

Projekt LarvixON dostarczył w pełni funkcjonalny **MVP (Minimum Viable Product)** składający się z pięciu zintegrowanych elementów (*Backend, Frontend, ML Model, Symulacja, Serwis pacjentów*). Architekturę systemu przedstawiono na *Rysunku 1*.



Rysunek 1: Architektura systemu LarvixON – widok kontenerów

3.1 Przebieg implementacji modułów projektu

Implementacja projektu LarvixON wymagała równoległej pracy nad pięcioma repozytoriami i integracją między nimi. Poniżej przedstawiono przebieg prac i kluczowe decyzje techniczne dla każdego z modułów.

3.1.1 Larvixon Backend (Django)

Prace nad backendem rozpoczęto od stworzenia szkieletu **Django REST Framework**, koncentrując się na bezpieczeństwie i skalowalności. Kluczową decyzją było przyjęcie **PostgreSQL** jako bazy danych dla środowiska produkcyjnego (wdrożonego w **Azure App Service**), ze względu na jej stabilność i zaawansowane funkcje. Najważniejsze funkcjonalności to zarządzanie profilami użytkowników i system śledzenia zleconych analiz wideo, w którym każde badanie jest ściśle powiązane z rekordem pacjenta pobieranym z zewnętrznego serwisu FHIR.

3.1.2 Larvixon App (Flutter)

Interfejs użytkownika został zrealizowany przy użyciu frameworku **Flutter**, co gwarantuje działanie aplikacji na wielu platformach (*web*, *mobile*, *desktop*). Podstawowym założeniem architektonicznym było zastosowanie **Clean Architecture**, aby zapewnić łatwą testowalność i separację warstw logiki, prezentacji oraz danych. Proces implementacji obejmował zaprojektowanie spójnego interfejsu umożliwiającego przesyłanie wideo do analizy oraz interakcję z wynikami. Dodatkowo skonfigurowano podmoduł **Unity** w celu włączenia symulacji do ekosystemu.

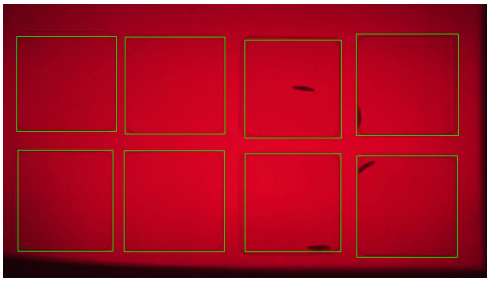
3.1.3 ML Model (PyTorch)

Implementacja modelu uczenia maszynowego przebiegała w **PyTorch**. Wybrano architekturę **hybrydową (CNN+LSTM)** jako optymalną do analizy danych czasowo-przestrzennych. Warstwa *CNN* (bazująca na *ResNet18*) służyła do ekstrakcji cech przestrzennych z pojedynczych klatek, podczas gdy warstwa *LSTM* odpowiadała za modelowanie dynamiki behawioralnej w sekwencji. Kluczowe zadania inżynierskie obejmowały: definicję formatu wejściowego (sekwencja N klatek o wymiarach $[N, 3, 112, 112]$), opracowanie *pipeline'u* predykcyjnego oraz konteneryzację modelu (Docker).

Model trenowano na danych przygotowanych przez Uniwersytet Medyczny we Wrocławiu. Zdefiniowano problem jako **klasyfikację wieloklasową** (*multi-class classification*), gdzie sieć uczyła się rozróżniać sygnatury ruchowe dla trzech odrębnych kategorii: ekspozycji na **etanol**, roztwór **kofeiny i tauryny** (napój energetyczny) oraz **wody** (próba kontrolna).

Preprocessing wideo i segmentacja szalek Ten etap koncentruje się na przekształceniu surowego pliku wideo w ustandaryzowany ciąg danych wejściowych gotowy dla modelu głębokiego uczenia. Zaznaczone przez model kontury szalek widać na *Rysunku 2*.

- **Wejście:** Surowy plik wideo przedstawiający larwy w komorze testowej.
- **Detekcja i śledzenie szalek:** Algorytm wizyjny identyfikuje i śledzi do **ośmiu osobnych szalek** na klatce. Proces ten obejmuje wydzielenie prostokątnego obszaru zainteresowania (*ROI*) wokół każdej szalki w każdej klatce.
- **Ekstrakcja sekwencji klatek:** Dla każdej z ośmiu szalek tworzona jest niezależna sekwencja klatek. Każda sekwencja jest normalizowana pod kątem wielkości (*np.* do 112×112 pikseli) i liczby klatek N (czas trwania obserwacji), tworząc tensor wejściowy o wymiarach $[N, 3, 112, 112]$.
- **Wyjście:** Zbiór ustandaryzowanych sekwencji wideo, gdzie każda sekwencja reprezentuje zachowanie jednej larwy.

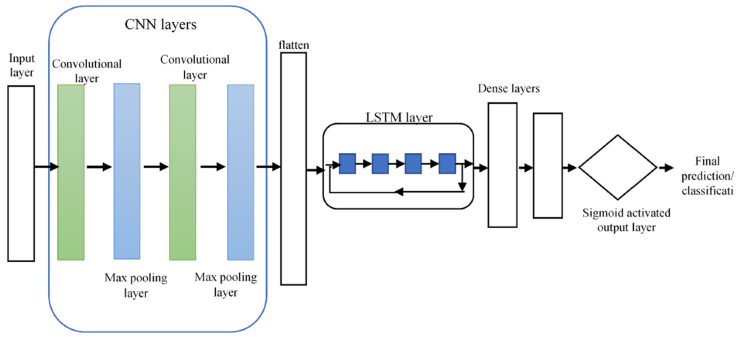


Rysunek 2: Detekcja naczyń, na których znajdowały się larwy

Architektura hybrydowa i ekstrakcja cech Tak przygotowane sekwencje danych są podawane do hybrydowej architektury modelu $CNN + LSTM$ w celu ekstrakcji cech czasowo-przestrzennych [*Rysunek 3.*]

Konwolucyjna sieć neuronowa (CNN) Każda pojedyncza klatka jest niezależnie przetwarzana przez pretrenowaną sieć **CNN** (np. $ResNet18$). Sieć uczy się rozpoznawać statyczne cechy ruchu, kształtu i pozycji larwy w danej chwili. Wyjściem dla każdej klatki jest wektor cech wysokiego poziomu (*embedding*), pozbawiony informacji o kolejności.

Długoterminowa pamięć krótkotrwała (LSTM) Sekwencja wektorów cech, wygenerowana przez warstwę CNN, jest podawana do sieci **LSTM**. Model ten uczy się, jak wzorce ruchu (*np.* skurcze, spowolnienie, zmiana kierunku) rozwijają się w kolejnych klatkach, śledząc zależności i kontekst w czasie. Wyjściem $LSTM$ jest finalny wektor stanu, zawierający skondensowaną informację o **całym wzorcu behawioralnym**.



Rysunek 3: Struktura modelu

3.1.4 Larvixon Simulation (Unity)

Moduł symulacyjny został zaimplementowany w środowisku **Unity**. Kluczowym wyzwaniem było stworzenie realistycznego modelu fizycznego **perystaltycznego ruchu larw** (5-segmentowa struktura) opartego na skurczach i rozciąganiu. Zaprogramowano mechanizm do modyfikacji parametrów ruchu w zależności od symulowanego działania substancji. Symulacja posłużyła do walidacji architektury sieci i debugowania pipeline'u, gdy nie było jeszcze dostępu do prawdziwych danych treningowych. Finalny model produkcyjny został wytrenowany wyłącznie na danych biologicznych.

3.1.5 Serwis Pacjentów (FastAPI)

Zaimplementowano również serwis dostarczający dane pacjentów do reszty aplikacji (na razie dane syntetyczne). Jednak dzięki zgodności z globalnym standardem **FHIR** *(Fast Healthcare Interoperability Resources)*, podłączenie aplikacji do realnych, szpitalnych serwisów w przyszłości będzie znacznie ułatwione.

3.2 Osiągnięte cele techniczne i biznesowe

Przy realizacji projektu udało się osiągnąć następujące cele:

- **Cel techniczny ogólny:** Sukcesywnie zaimplementowano złożoną architekturę wielomodułową z pełnym wsparciem dla **konteneryzacji (Docker)** w każdym repozytorium (*backend, ML*), co zapewnia łatwe wdrożenie, przenośność i skalowalność. Wdrożenie na środowisko chmurowe zostało zakończone, wykorzystując **Microsoft Azure App Service** z bazą danych *PostgreSQL*.
- **Cel techniczny – Model ML:** Pomyślne wdrożenie i walidacja **hybrydowego modelu klasyfikacyjnego CNN+LSTM**, zdolnego do przetwarzania sekwencji klatek wideo i klasyfikowania wzorców behawioralnych z dokładnością w przedziale 60%–80%, co udowadnia siłę predykcyjną behawioru larw.
- **Cel techniczny – zmniejszenie ryzyka braku danych:** Osiągnięto działający moduł **Symulacji Unity**, który pomyślnie rozwiązał początkowy problem braku wystarczającej ilości rzeczywistych danych biologicznych poprzez generowanie dużych ilości wiarygodnych, oznaczonych wideo do wstępnego treningu modelu *ML*.

- **Cel biznesowy ogólny:** Zaprojektowano działający prototyp diagnostyczny, potwierdzający, że połączenie biologicznego modelu *G. mellonella* z głębokim uczeniem jest technicznie wykonalne i ma **potencjał komercyjny** jako narzędzie do szybkiego screeningu toksykologicznego.
- **Cel biznesowy – integracja systemowa:** Dzięki integracji z serwisem pacjentów, który jest zgodny ze standardem medycznym **FHIR (Fast Healthcare Interoperability Resources)**, wdrożenie systemu do użytku szpitalnego i jego komunikacja z elektroniczną dokumentacją medyczną (*EDM*) są znacznie ułatwione i przyspieszone.
- **Cel biznesowy – efektywność kosztowa:** Osiągnięcie zdolności do wstępnego screeningu o obiecującej przepustowości oznacza, że **koszt jednostkowy** testu toksykologicznego jest znacząco niższy niż w przypadku standardowej chromatografii, co otwiera drogę do szerokiego zastosowania w diagnostyce przesiewowej.

4 WNIOSKI

Projekt LarvixON AI został pomyślnie zrealizowany, osiągając wszystkie kluczowe cele techniczne i biznesowe. Stanowi on dowód na możliwość wykorzystania algorytmów głębokiego uczenia w połączeniu z innowacyjnym modelem biologicznym do celów diagnostycznych.

4.1 Podsumowanie i znaczenie wyników

Najważniejszy sukces projektu polega na technicznej walidacji koncepcji integracji zaawansowanych algorytmów AI z modelem biologicznym. Dzięki temu z sukcesem stworzono system, który stanowi obiecujący, **skalowalny i ekonomicznie efektywny** prototyp narzędzia do wstępnego screeningu toksykologicznego. Opracowane rozwiązanie ma bezpośrednie znaczenie dla docelowej grupy odbiorców, oferując szybką alternatywę dla drogich badań laboratoryjnych.

Osiągnięty poziom dokładności klasyfikacji substancji przez model CNN+LSTM waha się w przedziale od 60 do 80 procent, w zależności od klasy ksenobiotyku oraz jakości przetwarzanych danych. Ta metryka jest kluczowa dla oceny praktycznej użyteczności całego systemu LarvixON.

4.2 Interpretacja osiągniętej dokładności (60-80%)

- **Wykazanie feasibility (wykonalności):** Wyniki potwierdzają **techniczną wykonalność** i skuteczność połączenia analizy behawioralnej *G. mellonella* z algorytmami głębokiego uczenia. Poziom dokładności znacznie przekraczający losowe zgadywanie (*baseline*) potwierdza wartość predykcyjną ze wzorców ruchowych.
- **Cel wstępnego screeningu:** Taka dokładność jest w pełni wystarczająca dla głównego celu projektu, czyli wstępnego screeningu diagnostycznego (pre-screening). System ma za zadanie szybko zawęzić listę potencjalnych substancji, redukując koszty i czas potrzebny na droższe, definitywne badania chemiczne.

4.3 Znaczenie praktyczne

LarvixON ustanawia nową, **szybką i ekonomiczną ścieżkę do wstępnego screeningu toksykologicznego**.

- **Wartość diagnostyczna:** Osiągnięta dokładność jest wystarczająca do pełnienia roli narzędzia **pre-screeningu**, redukującego potrzebę natychmiastowego sięgania po drogie metody chemiczne.
- **Kierunki dalszych prac:** Głównym zadaniem jest teraz podniesienie dokładności klasyfikacji (powyżej 80%) poprzez intensywne pozyskiwanie realnych danych klinicznych oraz dalszą optymalizację algorytmów. Kolejnym krokiem jest komercyjna walidacja i rozszerzenie bazy klasyfikowanych ksenobiotyków.

5 KIERUNKI ROZWOJU

W celu komercjalizacji i zwiększenia wartości projektu, w przyszłości należy podjąć następujące kierunki rozwoju:

- **Walidacja kliniczna i zasilenie danych:** Przeprowadzenie badań z większą ilością danych biologicznych in vivo w celu poprawy precyzji modelu ML oraz uzyskanie niezbędnych certyfikacji dla wdrożeń klinicznych.
- **Rozszerzenie bazy substancji:** Powiększenie zbioru klasyfikowanych ksenobiotyków i opracowanie systemu dynamicznego zarządzania nowymi substancjami bez konieczności re-treningu całego modelu.
- **Analiza w czasie rzeczywistym:** Ulepszenie modułów wideo i ML w celu wspierania strumieniowej analizy w czasie rzeczywistym, co przyspieszyłoby proces diagnostyczny.
- **Komercjalizacja i model biznesowy:** Opracowanie strategii wdrożenia produktu w modelu *Software as a Service (SaaS)* oraz jego integracji z istniejącymi systemami laboratoryjnymi (LIMS).
- **Integracja z centralnym systemem tożsamości (SSO/LDAP):** Implementacja mechanizmu logowania z wykorzystaniem szpitalnej bazy użytkowników. Pozwoli to na automatyzację zarządzania dostępem, wyeliminuje konieczność tworzenia osobnych kont dla personelu oraz zwiększy bezpieczeństwo poprzez centralną weryfikację uprawnień.
- **Dodatkowe funkcjonalności aplikacji:** Możliwość analizy wielu nagrań larw nastrzykniętych osoczem jednego pacjenta, w celu zwiększenia celności analizy i pewności jej wyniku

6 PODZIĘKOWANIA

Chcielibyśmy wyrazić swoją wdzięczność:

- Naszej opiekun ZPI dr inż. Natalii Piórkowskiej za wsparcie, ekspertyzę i zarządzanie naszą pracą.
- Naszemu mentorowi, dr inż. Marcinowi Jodłowcowi za merytorykę, cenne uwagi i pomoc techniczną przy powstawaniu projektu.
- Uniwersytetowi Medycznemu we Wrocławiu za udostępnienie danych do trenowania modeli, a także za informację zwrotną.
- Wrocławskiemu Centrum Sieciowo-Superkomputerowemu za udostępnienie sprzętu do trenowania modeli.

BIBLIOGRAFIA

[1] Everaerts Claude Grillet Micheline Ferveur Jean-François. „Data from: Behavioral elements and sensory cues involved in sexual isolation between Drosophila melanogaster strains". Angielski. W: *Przegląd Toksykologii i Medycyny Sądowej* (2018), s. 45–62.

[2] Chen Ningning Teh Noranis Mohd Aris. „Integration of CNN and LSTM Networks for Behavior Feature Recognition: An Analysis". W: *Proceedings of the International Conference on Deep Learning in Computer Vision (ICDLCV)*. IJASEIT, 2024, s. 112–120.

[3] Robert C. Martin. *Clean Architecture: A Craftsman's Guide to Software Structure and Design*. Prentice Hall, 2017. ISBN: 978-0134494166.

[4] Smołka Jakub Białkowski Damian. „Ocena wydajności czasowej frameworku Flutter w kontekście obsługi interfejsów użytkownika". W: *Inżynieria Oprogramowania Nowej Generacji* 8.1 (2022).

[5] The Django REST Framework Developers. *Documentation: Authentication and Permissions*. `https://www.django-rest-framework.org/api-guide/authentication/`. Dostęp: Grudzień 2025. 2024.

[6] Microsoft Azure. *Deploying Python applications to Azure App Service using containers*. `https://docs.microsoft.com/en-us/azure/app-service`. Dostęp: Grudzień 2025. 2024.

[7] Docker Inc. *Best practices for building images with Docker*. `https://docs.docker.com/develop/develop-images/dockerfile_best-practices/`. Dostęp: Grudzień 2025. 2024.

[8] Unity Technologies. *Unity Manual*. `https://docs.unity3d.com/Manual`. Dokumentacja techniczna, Dostęp: Grudzień 2025. 2020.