

**AKADEMIA GÓRNICZO-HUTNICZA**

WYDZIAŁ FIZYKI I INFORMATYKI STOSOWANEJ  
KIERUNEK INFORMATYKA STOSOWANA



PODSTAWY TWORZENIA APLIKACJI W OPARCIU O USŁUGI AZURE

---

## Dokumentacja projektu

---

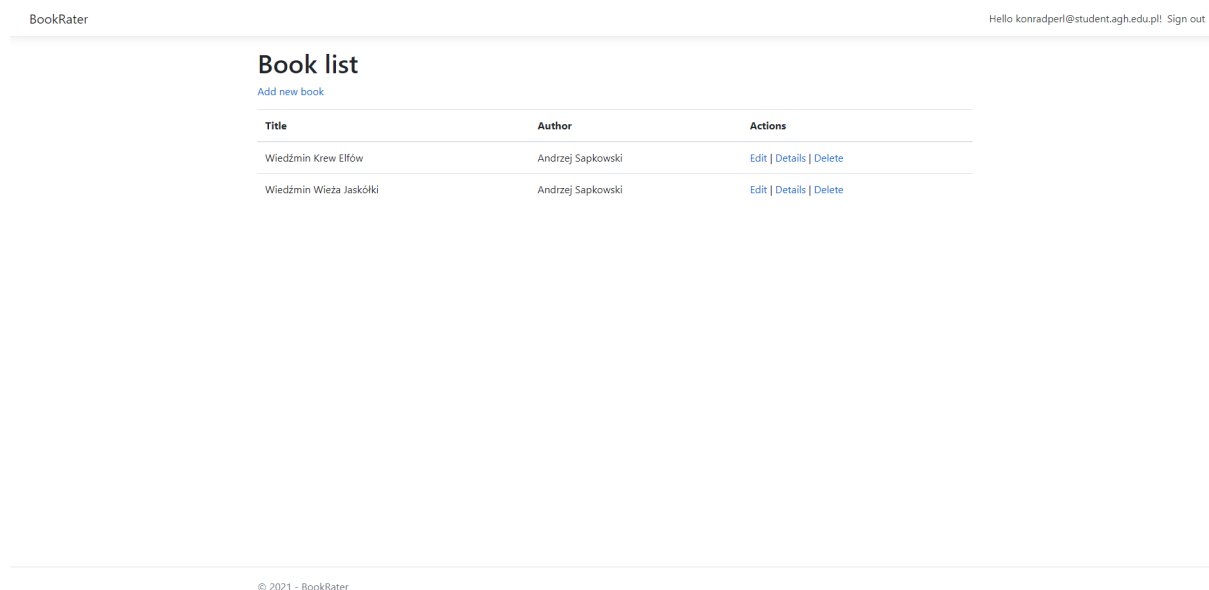
Konrad Perłowski

Kraków, 11 stycznia 2022 r.

# 1 Wstęp

Dokumentacja dotyczy projektu *BookRater* stworzonego jako projekt z przedmiotu Podstawy tworzenia aplikacji w oparciu o usługi Azure. Działający projekt znajduje się pod adresem: [link](#). kod źródłowy można znaleźć na repozytorium w serwisie github pod adresem: [link](#). Celem projektu było stworzenie aplikacji internetowej korzystającej z usług Azure.

Aplikacja *BookRater* pozwala na dodawanie i ocenianie książek. Przy dodawaniu książki wymagane jest podanie jej tytułu, autora oraz wgranie zdjęcia okładki. Wszystkie dodane książki znajdują się na stronie startowej. Ocena danej książki jest możliwa po wejściu w jej szczegóły, gdzie również możemy znaleźć średnią jej ocenę. Każdy z odwiedzających stronę będzie proszony o zalogowanie poprzez konto Microsoft. Wygląd aplikacji został przedstawiony na rysunku 1.

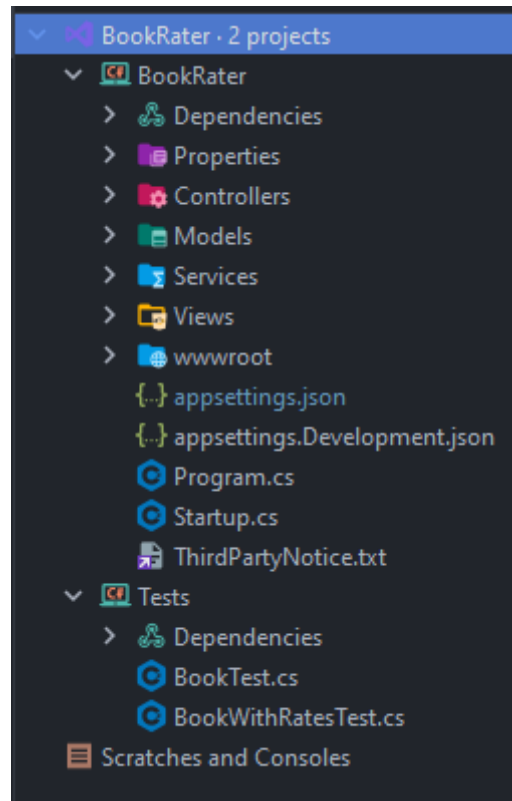


Rysunek 1: Strona główna aplikacji

## 2 Struktura

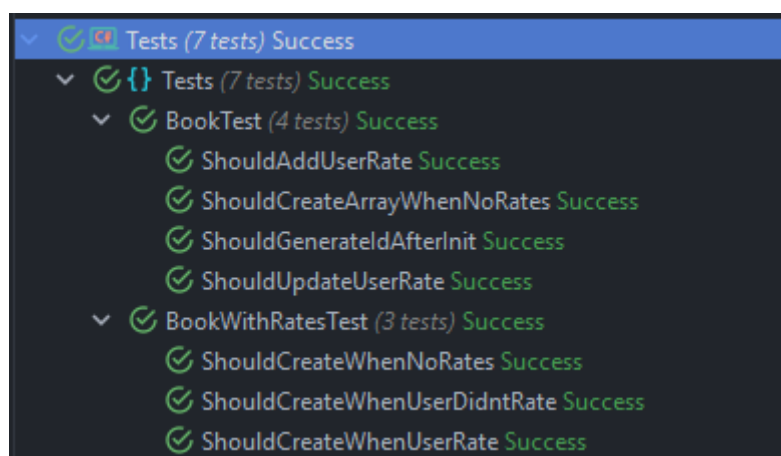
Aplikacja *BookRater* jest aplikacją napisaną w języku C z wykorzystaniem Asp.Net Core w wersji 5, została stworzona w oparciu o model MVC. Składa się ona z dwóch solucji - BookRater stanowiącej główną część aplikacji, oraz Tests zawierającą testy jednostkowe weryfikujące poprawność działania programu.

Struktura katalogów aplikacji została przedstawiona na rysunku 2.



Rysunek 2: Struktura aplikacji

Aplikacja testuje poprawne działanie klas *Book* oraz *BookWithRatings*, wynik działania testów został przedstawiony na rysunku 3.



Rysunek 3: Wyniki testów jednostkowych

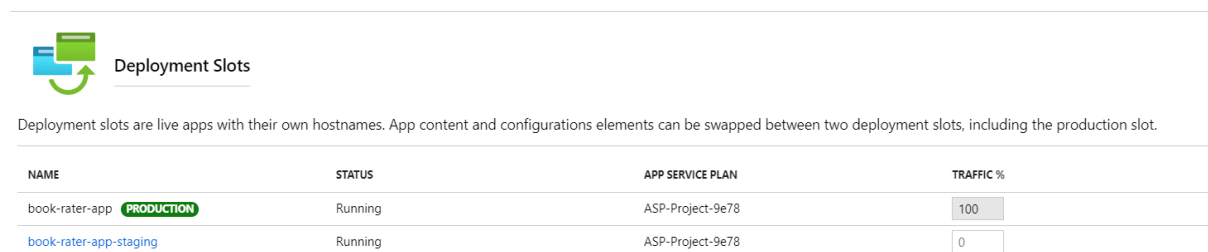
## 3 Komponenty Azure

### 3.1 WebApp

Projekt *BookRater* znajduje się na chmurze Azure. Sama aplikacja jest wdrożona jako Azure WebApp. Posiada ona dwa sloty - jeden o nazwie Production stanowiący wdrożoną główną aplikację na środowisku produkcyjnym oraz drugi slot staging - pozwalający na testowanie aplikacji przed jej wdrożeniem na produkcję.

Proces publikowania aplikacji został zautomatyzowany przy pomocy Github Actions. Projekt zawiera dwa osobne pliki .yml definiujące publikowanie aplikacji na zarówno środowisko testowe (staging) jak i produkcyjne (production). W obu przypadkach proces wygląda podobnie - przy pchnięciu zmian na brancha master lub staging uruchamiane są wszystkie testy aplikacji. Po ich poprawnym przejściu następuje faza budowania aplikacji zakończona publikacją kodu na odpowiedni deployment slot. Aplikacje w środowisku testowym można znaleźć pod adresem: link.

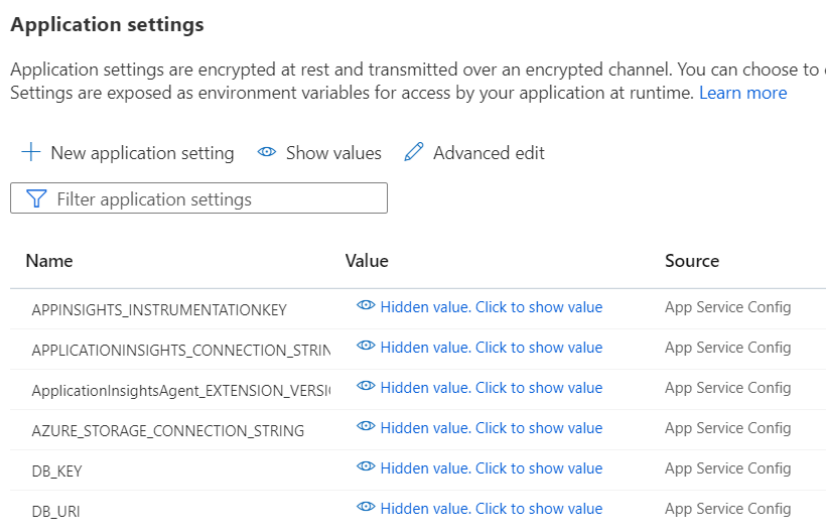
Aby skonfigurować środowisko testowe, nasz Azure WebApp musi być w co najmniej standardowym pakiecie. Wybrany w projekcie pakiet generuje koszty o wartości 73 dolary na miesiąc.



Deployment Slots			
Deployment slots are live apps with their own hostnames. App content and configurations elements can be swapped between two deployment slots, including the production slot.			
NAME	STATUS	APP SERVICE PLAN	TRAFFIC %
book-rater-app <b>PRODUCTION</b>	Running	ASP-Project-9e78	100
book-rater-app-staging	Running	ASP-Project-9e78	0

Rysunek 4: Prezentacja deployment slotów w panelu Azure

Konfiguracja aplikacji opiera się na użyciu ApplicationSettings w zakładce Configuration, gdzie ustawione są wszystkie niezbędne zmienne środowiskowe do działania aplikacji takie jak parametry dostępu do baz danych.

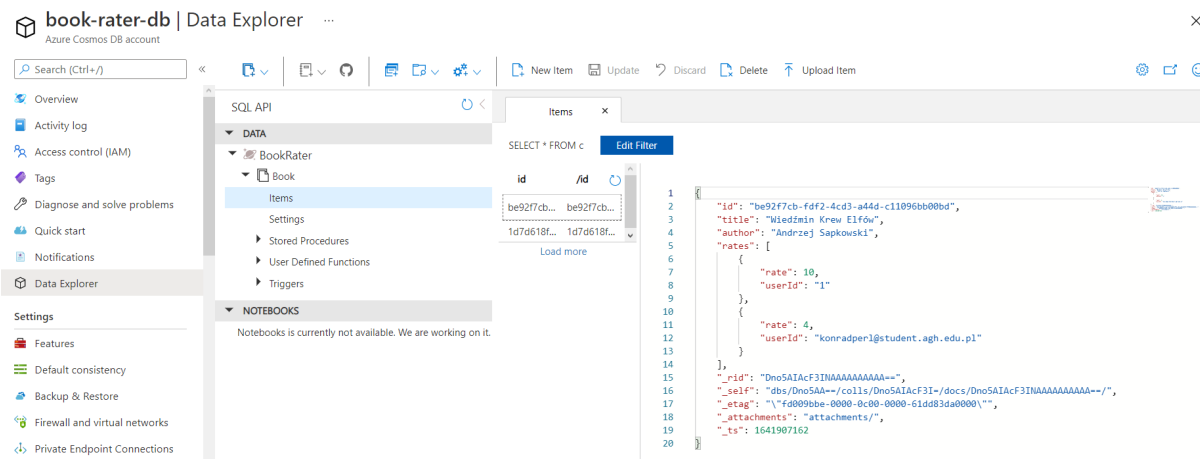


Application settings		
Application settings are encrypted at rest and transmitted over an encrypted channel. You can choose to Settings are exposed as environment variables for access by your application at runtime. <a href="#">Learn more</a>		
+ New application setting    👁 Show values    ✎ Advanced edit		
🔍 Filter application settings		
Name	Value	Source
APPINSIGHTS_INSTRUMENTATIONKEY	👁 Hidden value. Click to show value	App Service Config
APPLICATIONINSIGHTS_CONNECTION_STRING	👁 Hidden value. Click to show value	App Service Config
ApplicationInsightsAgent_EXTENSION_VERSION	👁 Hidden value. Click to show value	App Service Config
AZURE_STORAGE_CONNECTION_STRING	👁 Hidden value. Click to show value	App Service Config
DB_KEY	👁 Hidden value. Click to show value	App Service Config
DB_URI	👁 Hidden value. Click to show value	App Service Config

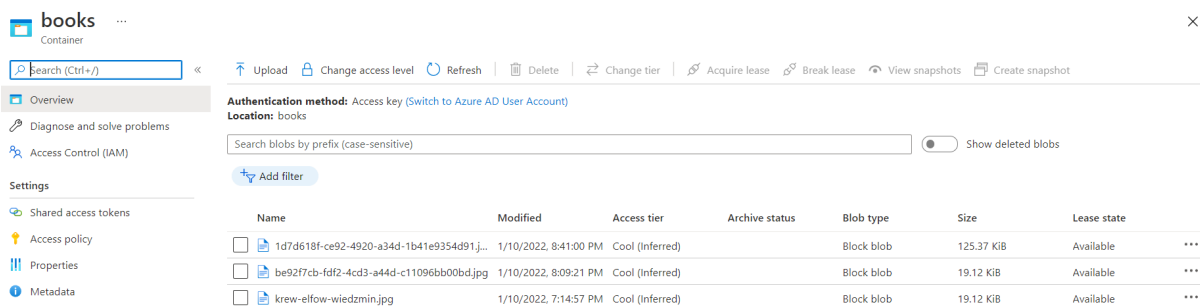
Rysunek 5: Prezentacja konfiguracji aplikacji

## 3.2 Bazy danych

Aplikacja do poprawnego działania wymaga zapis danych do trwałego magazynu danych. Musi zapisywać zarówno dane o książkach jak i zdjęcie jej okładki. W tym celu wykorzystane zostały dwie bazy - CosmosDB oraz Blob Storage. W tym pierwszym przechowywane są informacje o tytule i autorze książki, ale także o jej ocenach przez poszczególnych użytkowników. W Blob Storage przechowywane są wszystkie dodane zdjęcia okładek książek.



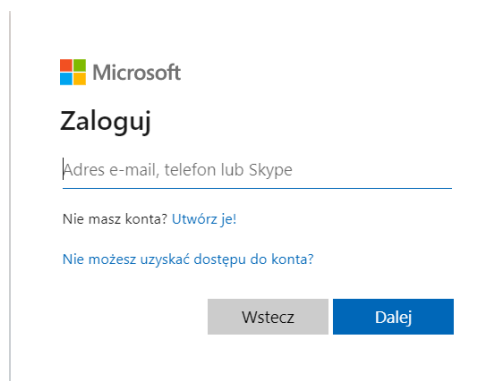
Rysunek 6: Prezentacja danych w bazie CosmosDB



Rysunek 7: Prezentacja danych w Blob Storage

### 3.3 Autoryzacja

Aby korzystać z aplikacji wymagane jest poprawne zalogowanie przy użyciu konta Microsoft - autoryzacja została zrealizowana przy użyciu Azure Active Directory. Po próbie wejścia na stronę użytkownik zostaje automatycznie przekierowany na stronę logowania Microsoft.

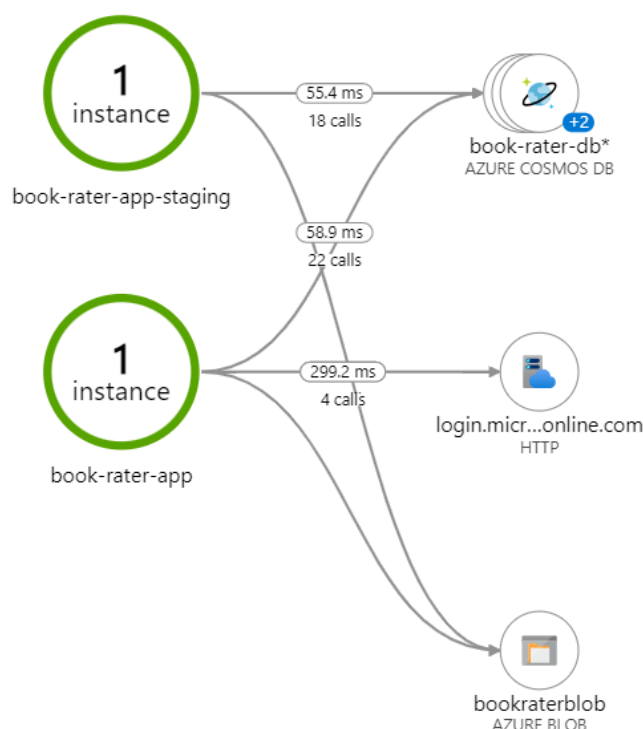


Rysunek 8: Strona logowania Microsoft

Dopiero po poprawnym zalogowaniu użytkownik ma możliwość przeglądania i oceniania książek.

### 3.4 Metryki i alerty

Do projektu została dodana paczka Application Insights pozwalająca na wszelkiego rodzaju analizę aplikacji. Dzięki zakładce Application Map jesteśmy w stanie zobaczyć w wizualny sposób strukturę projektu wraz z natężeniem ruchu pomiędzy poszczególnymi komponentami. Mapa projektu została przedstawiona na rysunku 9.



Rysunek 9: Mapa natężenia projektu

Na rysunku widać zarówno instancje produkcyjną jak i testową WebApp (Seksja 3.1), obie bazy danych: CosmosDB i Blob Storage (Seksje 3.2) jak i komponent odpowiadający za autoryzację - Azure Active Directory (Seksja 3.3).

Dużą korzyścią komponentu Application Insights jest ustawienie alertów. W projekcie zostało skonfigurowane automatyczne wysyłanie wiadomości e-mail w momencie wystąpienia jakiegokolwiek odpowiedzi 5xx. Konfiguracja alertu została pokazana na rysunku 10, a odebrana wiadomość e-mail na rysunku 11.

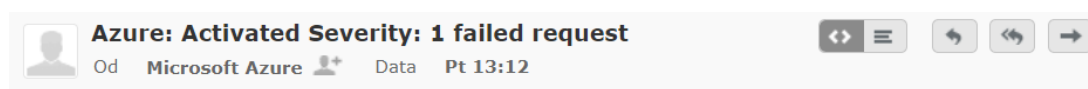
## Condition

Configure when the alert rule should trigger by selecting a signal and defining its logic.

### Condition name

✓ Whenever the count of failed requests is greater than 1

Rysunek 10: Konfiguracja alertu



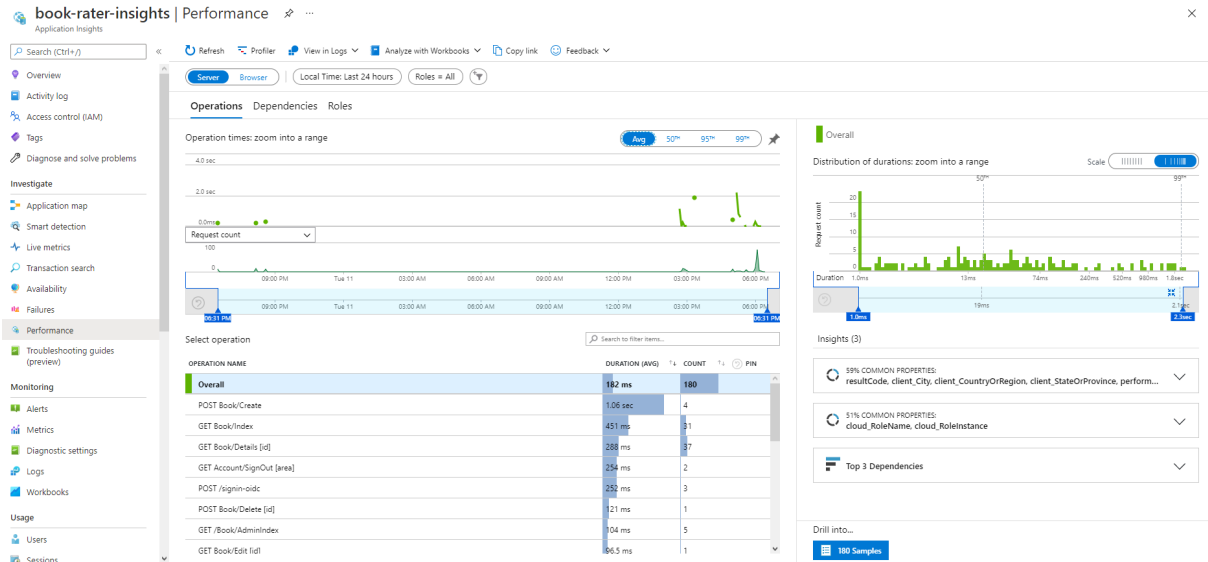
## ⚠ Your Azure Monitor alert was triggered

Azure monitor alert rule failed request was triggered for book-rater-insights at January 7, 2022 12:12 UTC.

Rule ID	/subscriptions/e18dfc30-2e4c-4924-811b-4812ea6535e0/resourceGroups/book-rater/providers/microsoft.insights/metricAlerts/failed request <a href="#">View Rule &gt;</a>
Resource ID	/subscriptions/e18dfc30-2e4c-4924-811b-4812ea6535e0/resourceGroups/book-rater/providers/microsoft.insights/components/book-rater-insights <a href="#">View Resource &gt;</a>

Rysunek 11: E-mail wywołany przez błąd 5xx

Poza alertami projekt bada również działanie aplikacji - dzięki wykorzystaniu Application Insights mamy dostęp do wielu różnych metryk działania aplikacji. W zakładce Performance możemy badać średni czas odpowiedzi oraz średnią ilość zapytań w danej godzinie. Poza tym w zakładce jest również przedstawiony średni czas dla konkretnych zapytań, ich łączna ilość wywołań. Pozwala to na dokładną analizę wydajności naszej aplikacji i łatwe znalezienie wolniej działających części kodu naszego projektu. Zakładka Performance przedstawiona została na rysunku 12.



Rysunek 12: Performance