Sigurnosna implementacija i integracija MERN stack aplikacije u Microsoft Azure

Jadrić, Mia

SVEUČILIŠTE U SPLITU

PRIRODOSLOVNO MATEMATIČKI FAKULTET

ZAVRŠNI RAD

SIGURNOSNA IMPLEMENTACIJA I INTEGRACIJA MERN STACK APLIKACIJE U MICROSOFT AZURE

Mia Jadrić

Split, srpanj 2024

Temeljna dokumentacijska kartica

Završni rad

Sveučilište u Splitu
Prirodoslovno-matematički fakultet
Odjel za informatiku
Ruđera Boškovića 33, 21000 Split, Hrvatska

Sigurnosna implementacija i integracija MERN stack aplikacije u Microsoft Azure

Mia Jadrić

SAŽETAK

Cilj ovog rada je opisati sigurnosnu implementaciju i integraciju MERN stack aplikacije u Microsoft Azure, istražiti različite aspekte sigurnosnih zahtjeva i strategija te primijeniti ih na praktičnom primjeru razvoja aplikacije u navedenom okruženju.

Ključne riječi: Microsoft Azure, MERN stack, računarstvo u oblaku, sigurnost u oblaku, Web Application Firewall, Web Application Gateway, Microsoft Sentinel

Rad sadrži: 54 stranice, 49 slika i 9 literaturnih navoda.

Mentor: **prof. dr. sc. Saša Mladenović**, *redoviti profesor Prirodoslovno-matematičkog fakulteta, Sveučilište u Splitu*

Ocjenjivači: **prof. dr. sc. Saša Mladenović**, *redoviti profesor Prirodoslovno-matematičkog fakulteta, Sveučilište u Splitu*
doc. dr. sc. Divna Krpan, *docent Prirodoslovno-matematičkog fakulteta, Sveučilište u Splitu*
doc. dr. sc. Goran Zaharija, *docent Prirodoslovno-matematičkog fakulteta, Sveučilište u Splitu*

Basic documentation card

Thesis

University of Split
Faculty of Science
Department of Informatics
Ruđera Boškovića 33, 21000 Split, Croatia

Security Implementation and Integration of a MERN Stack Application in Microsoft Azure

Mia Jadrić

SUMMARY

The aim of this thesis is to describe the security implementation and integration of a MERN stack application in Microsoft Azure, explore various aspects of security requirements and strategies, and apply them to a practical example in the mentioned environment.

Keywords: Microsoft Azure, MERN stack, cloud computing, cloud security, Web Application Firewall, Web Application Gateway, Microsoft Sentinel

Thesis consists of: 54 pages, 49 figures, and 9 references.

Mentor: **prof. dr. sc. Saša Mladenović**, *full professor at the Faculty of Science and Mathematics in Split, University of Split*

Reviewers: **prof. dr. sc. Saša Mladenović**, *full professor at the Faculty of Science and Mathematics in Split, University of Split*
Divna Krpan, Ph.D., *Assistant professor at the Faculty of Science and Mathematics in Split, University of Split*
Goran Zaharija, Ph.D., *Assistant professor at the Faculty of Science and Mathematics in Split, University of Split*

IZJAVA

kojom izjavljujem s punom materijalnom i moralnom odgovornošću da sam završni rad s naslovom „SIGURNOSNA IMPLEMENTACIJA I INTEGRACIJA MERN STACK APLIKACIJE U MICROSOFT AZURE“ izradila samostalno. U radu sam koristila literaturu koja je navedena na kraju završnog rada. Sve tuđe spoznaje, stavove, zaključke, teorije i zakonitosti koje sam citirala ili parafrizirala iz druge literature jasno sam označila citatima s navedenim izvorom iz kojeg su preneseni.

Studentica:

Mia Jadrić

Zahvaljujem se profesoru Saši Mladenoviću na pruženoj prilici za mentorstvo, kao i mentorima tvrtke Eviden na pomoći i resursima koje su mi omogućili tijekom izrade završnog rada.

Sadržaj

Uvod

U današnjem digitalnom dobu, sigurnost podataka postaje ključni prioritet za organizacije koje razvijaju i održavaju aplikacije u okruženjima kao što je Microsoft Azure. Implementacija sigurnosnih mjera postaje imperativ kako bi se osigurala zaštita osjetljivih informacija i spriječile potencijalne prijetnje. Računarstvo u oblaku postaje sveprisutan koncept potaknut promicanjem usluga računarstva u oblaku te širokom dostupnošću takvih usluga. Tijekom proteklog desetljeća, računarstvo u oblaku evoluiralo je iz futurističke ideje u stvarnost, postajući ključna komponenta informacijske tehnologije. Iako računarstvo u oblaku nije nov koncept, njegova definicija ostaje izazovna zbog širokog spektra usluga koje obuhvaća te različitih percepcija među korisnicima. Ovaj koncept omogućuje pristup zajedničkim IT resursima na zahtjev diljem svijeta, s ciljem automatskog osiguravanja skalabilnih i ekonomičnih resursa u pravo vrijeme te smanjenja potrebe za njihovim upravljanjem. Računarstvo u oblaku obuhvaća tri osnovna modela usluga: softver kao uslugu (SaaS), platformu kao uslugu (PaaS) i infrastrukturu kao uslugu (IaaS), te može biti implementirano u privatnom ili javnom modelu. Rastući interes za računarstvo u oblaku potiče natjecanje među pružateljima usluga, što rezultira širokim spektrom usluga i koristi za klijente. Obzirom na raznolikost usluga računarstva u oblaku, donošenje informiranih odluka pri odabiru postaje ključno. Motivacija za ovu studiju proizlazi iz izazova s kojima se suočavaju aplikacije u digitalnom poslovanju, posebno u pogledu sigurnosti podataka. Ključno je razumjeti kako implementirati učinkovite sigurnosne mjere radi zaštite podataka i integriteta sustava. Ovaj rad pruža pregled strukture istraživanja, počevši od definicije sigurnosti u kontekstu informacijske tehnologije, preko pregleda ključnih aspekata sigurnosti i računarstva u oblaku, do analize arhitekture MERN stack aplikacije te njezine implementacije u Microsoft Azure okruženju. Poseban fokus stavljen je na sigurnosne mjere za zaštitu osjetljivih podataka, otkrivanje i prevenciju napada te sigurnosno praćenje i analizu. U kontekstu ovog istraživanja, važno je istaknuti nedavnu migraciju tvrtke Atos u Eviden koristeći resurse na Microsoft Azure portalu. Osim već postojećih sigurnosnih mjera, opisane su i alternative koje su mogle biti odabrane umjesto trenutno implementiranih. Ovo istraživanje je ključno radi boljeg ilustriranja mogućnosti i sigurnosnih aspekata, posebno s obzirom na različite konfiguracije koje ovise o korisničkim računima. Kroz detaljnu analizu simuliranih napada i praćenja logova identificirat će se ključni aspekti sigurnosti.

1 Računarstvo u oblaku

Računarstvo u oblaku pruža usluge temeljene na pretplati koje omogućavaju korisnicima pristup prostoru za pohranu podataka i računalnim resursima putem mreže. Uspoređujući ga s tradicionalnim lokalnim računalnim infrastrukturama, računarstvo u oblaku nudi više skalabilnosti, resursne efikasnosti i fleksibilnosti. Umjesto vlastitih fizičkih servera koji zahtijevaju održavanje, nadogradnje i upravljanje, omogućava korisnicima pristup različitim vrstama računalnih resursa prema potrebi, uz plaćanje samo za korištene resurse. Također, omogućava brzu implementaciju i skaliranje aplikacija te smanjuje potrebu za ulaganjem u skupe infrastrukture.

Ključne karakteristike usluga u oblaku:

- Skalabilnost: Omogućuje fleksibilno prilagođavanje prema korisnikovim potrebama. Kapaciteti se mogu smanjivati ili povećavati prema zahtjevima korisnika.
- Virtualizacija: Više korisnika može dijeliti iste fizičke resurse, što omogućuje efikasno iskorištavanje resursa.
- Samouslužnost: Korisnici mogu samostalno pristupiti računalu putem sučelja u oblaku, bez pomoći administratora.
- Mjerenje usluga: Omogućuje precizno mjerenje korištenja resursa i naplatu samo za korištene resurse.

Korištenje usluga u oblaku uklanja potrebu za infrastrukturom i održavanjem hardvera i softvera, a korisnici plaćaju samo ono što stvarno koriste. Omogućava pristup računalnim resursima putem interneta, što znači da korisnici mogu pristupiti svojim podacima ili aplikacijama s bilo kojeg uređaja s internetskom vezom. Prednost usluga u oblaku jest i brza implementacija, što omogućuje korisnicima brzo pokretanje novih aplikacija i smanjenje vremena i resursa potrebnih za tradicionalnu implementaciju.

1.1 Vrste oblaka

Postoje različite vrste oblaka koje se koriste za različite potrebe:

- Javni oblak: Resursi su dostupni javnosti putem interneta i njima upravlja treća strana (npr. Amazon Web Services, Microsoft Azure, Google Cloud Platform).
- Privatni oblak: Resursi se koriste isključivo od strane jedne organizacije. Privatni oblak može biti smješten u lokalnom podatkovnom centru ili kod vanjskog pružatelja usluga.
- Hibridni oblak: Kombinacija javnog i privatnog oblaka koja omogućuje prijenos podataka i aplikacija između dva okruženja.
- Višeoblaka: Korištenje usluga od više pružatelja oblaka, omogućujući organizacijama da koriste najbolje usluge koje svaki pružatelj nudi.

1.2 Modeli usluga u oblaku

Arhitektura računarstva u oblaku može se kombinirati s različitim modelima usluga kako bi se zadovoljile specifične potrebe korisnika i organizacija. Tri glavna modela usluga u oblaku su:

1. Infrastruktura kao usluga (IaaS):
    - Pružatelji oblaka nude virtualiziranu infrastrukturu kao uslugu, uključujući virtualne strojeve, mrežne resurse i pohranu podataka.
    - Korisnici imaju potpunu kontrolu nad operativnim sustavima i aplikacijama, dok davatelj oblaka održava fizičku infrastrukturu.

2. Platforma kao usluga (PaaS):
    - Pružatelji oblaka nude platformu za razvoj, testiranje i implementaciju aplikacija kao uslugu.
    - Korisnici se fokusiraju na razvoj aplikacija, dok se briga o infrastrukturi prepušta pružatelju usluge oblaka.

3. Softver kao usluga (SaaS):

- Korisnici pristupaju softveru putem interneta kao usluzi, bez potrebe za instalacijom i održavanjem lokalnih verzija.
- Pružatelj oblaka je odgovoran za održavanje, sigurnost i skalabilnost softvera.
- Opisana arhitektura varira ovisno o modelima usluga u oblaku i može se prilagoditi specifičnim potrebama i ciljevima organizacije.

1.3 Glavni pružatelji usluga računarstva u oblaku

U kontekstu cloud computinga, tri vodeća pružatelja usluga dominiraju tržištem: Amazon Web Services (AWS), Microsoft Azure i Google Cloud Platform (GCP). Svaki od ovih pružatelja nudi širok spektar usluga i alata prilagođenih različitim potrebama korisnika (Borra, 2024). AWS se ističe po svojoj raznolikosti usluga, uključujući integrirane značajke za računarstvo, pohranu i analitiku, s više od 200 usluga koje podržavaju različite poslovne potrebe. Azure se ističe svojom integracijom s Microsoftovim proizvodima, olakšavajući migraciju postojećih aplikacija i infrastrukture u oblak te pruža napredne alate za razvoj i analizu podataka. S druge strane, GCP se fokusira na brzinu i skalabilnost te pruža napredne alate za analizu podataka i strojno učenje (Borra, 2024).

Što se tiče sigurnosti i usklađenosti, sva tri pružatelja usluga imaju visoke standarde, ali se razlikuju u specifičnim sigurnosnim značajkama. AWS pruža širok spektar alata za upravljanje pristupom, enkripciju podataka i detekciju prijetnji, dok Azure ističe svoje integrirane alate za upravljanje identitetom i pristupom te napredne usluge zaštite podataka. GCP se ponosi svojom proaktivnom sigurnosnom politikom i visokom razinom transparentnosti u vezi s infrastrukturom i sigurnosnim praksama.

Svaki od pružatelja usluga također nudi alate za integraciju s postojećim sustavima, ali se razlikuju u razini podrške za različite tehnologije i platforme. AWS i Azure često se percipiraju kao skuplji od GCP-a, ali svaki ima svoj model cjenovne politike koji odgovara različitim potrebama korisnika. AWS i Azure nude kompleksne modele cijena koji se temelje na potrošnji resursa, dok GCP nudi jednostavniji pristup, često s jeftinijim cijenama (Borra, 2024).

Kada se uzmu u obzir specifične potrebe organizacije, težnje za integracijom s Microsoftovim ekosustavom, visoka razina sigurnosti i podrška za razne operativne sustave, uključujući Windows, Microsoft Azure se izdvaja kao optimalan izbor, pružajući široku paletu usluga u oblaku, fleksibilnost i skalabilnost uz napredne sigurnosne značajke. Međutim, za organizacije koje preferiraju agilnost, inovaciju i napredne alate za analizu podataka, GCP može biti privlačnija opcija, dok AWS ostaje popularan izbor za širok spektar poslovnih potreba i industrija (Borra, 2024).

1.4 Rizici računarstva u oblaku

Računarstvo u oblaku donosi mnoge prednosti, ali isto tako nosi određene rizike koje korisnici trebaju imati na umu prilikom korištenja ovih usluga. Jedan od ključnih rizika je sigurnost podataka. Iako vodeći pružatelji cloud usluga ulažu značajne napore u osiguravanje sigurnosti, postoji uvijek rizik od neovlaštenog pristupa, hakiranja ili gubitka podataka. To može rezultirati krađom osjetljivih informacija ili narušavanjem povjerljivosti podataka.

Drugi važan rizik je dostupnost usluga. Iako se cloud usluge obično smatraju pouzdanima, može doći do nepredviđenih prekida usluga zbog tehničkih problema, kvarova opreme ili čak napada hakera. To može rezultirati nedostupnošću aplikacija i podataka, što može imati negativne posljedice po poslovanje korisnika. Nadalje, korištenje cloud usluga može izazvati pitanja u vezi s pravnom odgovornošću i regulativama, posebno kada su u pitanju prijenos podataka između različitih zemalja ili pridržavanje specifičnih industrijskih standarda. Postoji i rizik od vendor lock-ina, što znači da korisnici mogu postati "zaključani" za određenu platformu, otežavajući migraciju aplikacija i podataka na drugog pružatelja usluga.

Dodatno, korisnici mogu osjećati gubitak kontrole nad svojim podacima kada se ti podaci pohranjuju i obrađuju na infrastrukturi treće strane, što može izazvati zabrinutost u vezi s privatnošću i povjerljivošću podataka. Troškovi su još jedan važan faktor, jer iako cloud computing može biti ekonomičan u početku zbog modela plaćanja prema korištenju, postoji rizik od nekontroliranih troškova ako se resursi ne upravljaju učinkovito ili ako se ne prate potrebe korisnika. Upravljanje identitetom i pristupom također su ključni za sigurno korištenje cloud usluga, pa nedostatak adekvatnih sigurnosnih mjera može dovesti do neovlaštenog pristupa korisničkim računima i podacima.

2 Sigurnost u oblaku

Sigurnost predstavlja kompleksan skup mjera i praksi usmjerenih na zaštitu podataka, sustava i korisnika od različitih prijetnji i rizika. Osnovni ciljevi sigurnosti uključuju zaštitu integriteta, povjerljivosti i dostupnosti podataka. U kontekstu računarstva u oblaku, sigurnost se proširuje na primjenu sigurnosnih mjera prilagođenih dinamičnoj prirodi oblaka, uzimajući u obzir skalabilnost, fleksibilnost i dinamičnost okruženja. Osnovni ciljevi sigurnosti uključuju zaštitu integriteta, povjerljivosti i dostupnosti podataka. U okruženju računarstva u oblaku, ovi ciljevi postaju još važniji s obzirom na dinamičnu prirodu oblaka. Jedan od glavnih razloga za naglašavanje sigurnosti u računarstvu u oblaku je zaštita osjetljivih podataka korisnika i tvrtki. Financijske informacije, osobni identifikacijski podaci, zdravstveni zapisi i druge osjetljive informacije često se pohranjuju u oblaku, stoga je ključno osigurati visoke standarde sigurnosti radi sprječavanja neovlaštenog pristupa, krađe ili gubitka. Osim toga, sigurnost u oblaku igra ključnu ulogu u očuvanju povjerenja korisnika. Korisnici očekuju da će njihovi podaci biti sigurni i privatni tijekom korištenja usluga u oblaku. Nedostatak odgovarajućih sigurnosnih mjera može narušiti to povjerenje i nanijeti štetu reputaciji tvrtke. Također, računarstvo u oblaku može biti meta raznih prijetnji poput DDoS napada, phishinga, ransomware-a i krađe identiteta. Implementacija sigurnosnih mjera ključna je za sprečavanje ovih napada i očuvanje integriteta sustava.

2.1 Ključni izazovi sigurnosti u oblaku

Sigurnost u oblaku suočava se s izazovima koji zahtijevaju pažljivo planiranje i implementaciju kako bi se osigurala zaštita podataka i sustava te uključuju sljedeće aspekte (Mather, T., Kumaraswamy, S., & Latif, S., 2009):

- Podatkovna privatnost i povjerljivost: Podaci pohranjeni u oblaku često uključuju osjetljive informacije poput financijskih podataka, osobnih identifikacijskih podataka i zdravstvenih zapisa. Ključni izazov je osigurati da ovi podaci ostanu privatni i zaštićeni od neovlaštenog pristupa.

- Sigurnost prijenosa podataka: Prijenos podataka između korisnika i cloud usluga predstavlja sigurnosni rizik. Osiguranje sigurnosti tijekom prijenosa može se postići korištenjem sigurnih komunikacijskih protokola kao što su TLS/SSL.

- Upravljanje pristupom: Upravljanje tko ima pristup kojim podacima i uslugama u oblaku je kritično za sprječavanje neovlaštenog pristupa. Implementacija robusnih rješenja za upravljanje identitetom i pristupom (IAM) može pomoći u postizanju ove kontrole.

- Usklađenost i regulative: Različite industrije i zemlje imaju specifične zakone i regulative koje definiraju kako se podaci moraju čuvati i obrađivati. Osiguravanje usklađenosti s ovim zakonima može biti kompleksno, posebno za globalne organizacije.

- Održavanje integriteta i dostupnosti: Integritet podataka osigurava se zaštitom od neovlaštenih izmjena. Osiguravanje da su podaci i resursi dostupni ovlaštenim korisnicima kada su potrebni je ključni aspekt sigurnosti. To uključuje otpornost na DDoS napade i druge prijetnje koje mogu ometati dostupnost usluga.

- Reakcija na incidente i opseg odgovornosti: Brzo i učinkovito odgovaranje na sigurnosne incidente ključno je za minimiziranje štete. U cloud okruženju, odgovornost za sigurnost dijele pružatelji usluga i korisnici, što zahtijeva jasno definirane ugovore i razumijevanje odgovornosti.

2.2 Najbolje prakse za sigurnost u oblaku

Najbolje prakse za sigurnost u oblaku obuhvaćaju širok spektar mjera i strategija koje su usmjerene na zaštitu podataka, sustava i korisnika (Mather, T., Kumaraswamy, S., & Latif, S., 2009):

- Šifriranje podataka tijekom prijenosa i mirovanja ključna je praksa za zaštitu povjerljivosti podataka. Korištenje jakih kriptografskih algoritama osigurava da čak i ako podaci budu kompromitirani, neće biti čitljivi neovlaštenim stranama.

- Implementacija robusnih rješenja za upravljanje identitetom i pristupom omogućuje preciznu kontrolu nad time tko ima pristup kojim podacima i uslugama u oblaku. To uključuje primjenu principa najmanje privilegije i redovito reviziju pristupa.

- Provođenje redovitih audita i procjena rizika ključno je za identifikaciju potencijalnih sigurnosnih propusta i ranjivosti u infrastrukturi oblaka. To omogućuje pravovremeno otkrivanje i otklanjanje sigurnosnih prijetnji.

- Implementacija sigurnosnih alata poput firewalla, IDS/IPS sustava (Intrusion Detection/Prevention Systems) i antivirusnih programa pomaže u zaštiti infrastrukture oblaka od vanjskih prijetnji. Integracija tih alata u sigurnosne arhitekture pruža dodatni sloj zaštite.

2.3 Pregled alata za sigurnost u Microsoft Azure okruženju

U Microsoft Azure okruženju postoji niz alata i usluga koji su namijenjeni osiguravanju sigurnosti podataka i infrastrukture.

1. Azure Security Center je integrirani alat za upravljanje sigurnosti koji pruža sveobuhvatni pregled sigurnosti vaših Azure resursa. Pruža preporuke za poboljšanje sigurnosti, upozorenja na potencijalne prijetnje te mogućnosti za automatsko otklanjanje sigurnosnih problema.

2. Azure Active Directory je usluga upravljanja identitetom koja omogućuje sigurnu autentikaciju i autorizaciju korisnika za pristup Azure uslugama i aplikacijama. Omogućuje jednostavno upravljanje pristupnim kontrolama i pruža napredne značajke za zaštitu identiteta.

3. Azure Key Vault je usluga za sigurno pohranjivanje tajnih ključeva, lozinki i drugih tajnih podataka. Pruža centralizirano upravljanje ključevima i omogućuje njihovo sigurno korištenje u aplikacijama i servisima u oblaku.

4. Azure Sentinel je usluga za sigurnosno informacijsko i događajno upravljanje koja pruža napredno otkrivanje prijetnji, analizu sigurnosnih događaja i odgovor na incidente. Integrira se s različitim izvorima podataka kako bi pružio cjelovit pregled sigurnosnih događaja u vašem Azure okruženju.

5. Azure Firewall je usluga za zaštitu mrežne sigurnosti koja pruža napredne sigurnosne funkcije poput filtriranja prometa, detekcije i prevencije prijetnji te zaštite od DDoS napada. Omogućuje centralizirano upravljanje sigurnosnim pravilima i kontrolama za zaštitu mrežnih resursa.

6. Azure DDoS Protection usluga pruža zaštitu od DDoS (Distributed Denial of Service) napada, sprječavajući preopterećenje mrežnih resursa i održavajući dostupnost aplikacija i usluga u Azure okruženju.

Ovi alati i usluge čine integrirani skup sigurnosnih alata u Microsoft Azure okruženju, pružajući naprednu zaštitu od različitih prijetnji i osiguravajući sigurnost podataka i infrastrukture u oblaku.

3 Arhitektura MERN stack aplikacije integrirane u Microsoft Azure

U ovom poglavlju istražit ćemo arhitekturu MERN stack aplikacije integrirane u Microsoft Azure okruženje. Analizirat ćemo ključne komponente ove arhitekture, usporediti je s drugim popularnim okvirima i alatima za razvoj web aplikacija, te objasniti zašto je odabrana MERN stack aplikacija za integraciju u Azure.

3.1 Arhitektura MERN stack aplikacije

MERN stack integrira MongoDB kao bazu podataka, Express.js za web server, React.js za korisničko sučelje te Node.js za izvođenje JavaScript koda na poslužiteljskoj strani. Ova kombinacija tehnologija omogućuje izgradnju full-stack web aplikacija i pruža programerima jedinstveno iskustvo korištenja samo jednog jezika, JavaScripta, za razvoj i front-end i back-end dijelova aplikacija. Jedna od ključnih prednosti MERN stoga je njegova sposobnost da podnese velike količine podataka i prometa bez gubitka performansi. Node.js, koji je temelj MERN-a, poznat je po visokoj stabilnosti i performansama, čineći ga idealnim za aplikacije koje trebaju obraditi velike količine podataka i visok promet. Ova kombinacija čini ga sveobuhvatnim stack-om tehnologija za izgradnju mobilnih i web aplikacija, koje se mogu integrirati s Microsoft Azureom za dodatnu pouzdanost i skalabilnost.

3.1.1 Opis MERN stack aplikacije integrirane s Microsoft Azureom

E-Banka je mini web aplikacija koja koristi MERN stack tehnologije kako bi pružila korisnicima bankarskih usluga i funkcionalnosti putem internetskog sučelja. Na korisničkom dijelu, aplikacija nudi korisnicima mogućnost jednostavne prijave ili registracije novog računa. Nakon prijave, korisnici dobivaju pristup svom osobnom bankovnom računu s detaljnim pregledom stanja i transakcija. Filtri omogućavaju precizno pretraživanje transakcija prema vremenskom periodu ili vrsti. Jedna od ključnih funkcionalnosti aplikacije je mogućnost izvršavanja prijenosa sredstava između korisničkih računa i internih prijenosa. Korisnici mogu

jednostavno odabrati račune između kojih žele obaviti prijenos, unijeti iznos i potvrditi transakciju.

Na pozadinskoj strani, aplikacija koristi autentikaciju i autorizaciju kako bi osigurala siguran pristup korisničkim računima i transakcijama. Također, pruža dohvat informacija o korisničkim računima, izvršava prijenose sredstava te omogućuje dohvat korisničkih podataka profila. Sve ove funkcionalnosti rade zajedno kako bi osigurale pouzdanu i funkcionalnu platformu za upravljanje financijskim transakcijama putem interneta.

3.2 Usporedba s drugim razvojnim stack-ovima

Kada uspoređujemo MERN stack s drugim popularnim razvojnim stackovima, važno je uzeti u obzir različite aspekte kao što su brzina razvoja, skalabilnost, performanse i podrška zajednice.

MEAN stack (MongoDB, Express.js, Angular.js, Node.js) je vrlo sličan MERN stacku, s glavnom razlikom u frontend frameworku (Angular.js umjesto React.js). Angular.js je cjeloviti frontend framework koji pruža sveobuhvatan set alata za izradu složenih aplikacija, dok React.js pruža veću fleksibilnost i komponentni pristup razvoju UI-a. MERN stack često dobija prednost zbog popularnosti i jednostavnosti React.js-a.

LAMP stack (Linux, Apache, MySQL, PHP/Python/Perl) je tradicionalni stack za izgradnju web aplikacija. Dok je vrlo pouzdan i ima dugu povijest, LAMP stack može biti manje učinkovit za moderne aplikacije koje zahtijevaju visoku skalabilnost i interaktivnost u stvarnom vremenu, što je prednost MERN stacka.

.NET stack koristi se za razvoj web aplikacija na platformi Microsoft, koristeći ASP.NET Core za backend i razne tehnologije poput C# za programiranje. Iako je .NET stack poznat po svojoj skalabilnosti i performansama, MERN stack nudi otvoreno-kodno rješenje s bogatom zajednicom i ekosustavom alata, što ga čini atraktivnim izborom za moderni web razvoj.

Ruby on Rails je framework koji koristi Ruby jezik i poznat je po svojoj jednostavnosti i brzini razvoja. Međutim, MERN stack može pružiti bolje performanse za aplikacije koje zahtijevaju visoku skalabilnost i obradu u stvarnom vremenu.

MERN stack se često smatra preferiranim izborom u usporedbi s ostalim tehnološkim stekovima zbog nekoliko ključnih prednosti. Prvo, MERN stack koristi JavaScript kao glavni jezik na svim razinama razvoja - od frontend-a (React.js) do backend-a (Node.js). Ova jednojezična paradigma smanjuje složenost projekta, olakšava komunikaciju između razvojnih timova i ubrzava proces razvoja i održavanja aplikacije. U usporedbi s MEAN stekom, koji koristi Angular umjesto Reacta, React.js se često smatra fleksibilnijim i jednostavnijim za učenje, posebno za manje i srednje velike projekte. U odnosu na LAMP stog, koji se oslanja na kombinaciju Linuxa, Apachea, MySQL-a i PHP-a, MERN stog nudi bolju skalabilnost kroz MongoDB i brže upravljanje korisničkim sučeljem kroz React.js. Nasuprot .NET steku, MERN stog pruža otvoreno-kodno rješenje s bogatom zajednicom i ekosustavom alata. Također, u usporedbi s Ruby on Rails, MERN stog nudi veću fleksibilnost u odabiru tehnologija i integracija. Sve ove prednosti čine MERN stog atraktivnim izborom za razvoj modernih web aplikacija koje zahtijevaju brzinu, skalabilnost, fleksibilnost i efikasno upravljanje korisničkim sučeljem.

4 Integracija aplikacije u Microsoft Azure

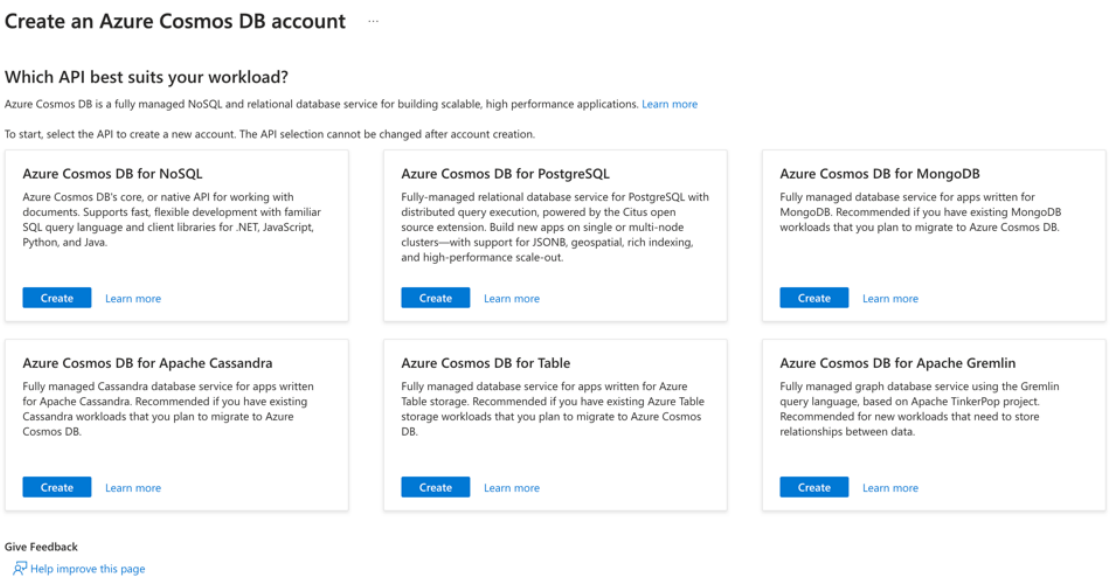
Integracija aplikacije u Microsoft Azure pruža mnoge prednosti, uključujući skalabilnost, pouzdanost i napredne usluge u oblaku. U ovom poglavlju, prikazat ćemo ključne korake i komponente procesa integracije aplikacije u Microsoft Azure te kako iskoristiti prednosti ove platforme.

4.1 Implementacija Azure Cosmos DB za MongoDB

Azure Cosmos DB predstavlja moćan alat za upravljanje NoSQL bazama podataka, a posebno za podršku MongoDB bazi podataka. U ovom dijelu prikazat ćemo proces implementacije Azure Cosmos DB-a kao zamjene za MongoDB, omogućujući skalabilnost, pouzdanost i učinkovitost u radu s podacima

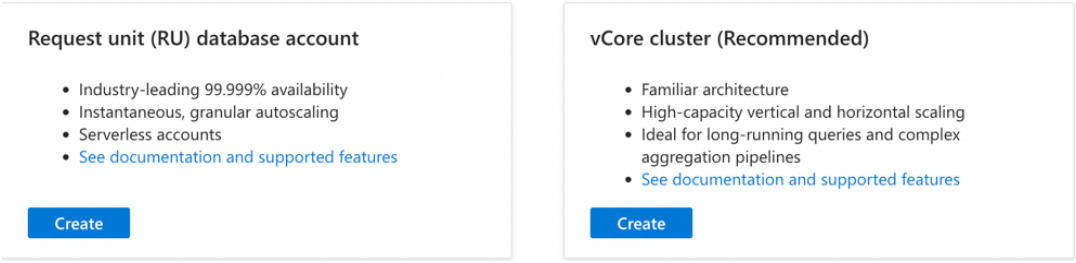
4.1.1 Kreiranje Azure Cosmos DB računa

Prvi korak je stvaranje Azure Cosmos računa putem Azure portala. Prilikom kreiranja računa, odaberemo MongoDB kao API koji želimo koristiti s Azure Cosmos DB-om.



Slika 1 Kreiranje Azure Cosmos računa – odabir API-ja

Azure Cosmos DB za MongoDB nudi dva osnovna modela resursa, svaki s jedinstvenom arhitekturom i prednostima koje pružaju.



Slika 2 Kreiranje Azure Cosmos računa - odabir između RU i vCore

Request Units su ključne jedinice za mjerenje i naplatu resursa potrebnih za operacije u Azure Cosmos DB-u za MongoDB, uključujući CPU, memoriju i I/O operacije.

Prednosti RU modela:

- Pruža precizno podešavanje propusnosti, što je idealno za aplikacije koje zahtijevaju dinamično skaliranje.
- Model je prikladan za cloud-native MongoDB aplikacije ili refaktoriranje postojećih, pružajući jednostavan pristup i skalabilnost.
- Omogućava beskonačno horizontalno skaliranje, što je ključno za rastuće i dinamične aplikacije.

vCore predstavlja virtualnu jedinicu procesora koja određuje računalne resurse potrebne za operacije u bazi podataka. Ovaj model se usredotočava na fiksne naknade temeljene na računalnim resursima i prostoru za pohranu.

Prednosti vCore modela:

- Prikladan je za aplikacije s dugotrajnim upitima, složenim agregacijama i transakcijama.
- Pruža vertikalno i horizontalno skaliranje na visokoj razini kapaciteta, idealno za velike i zahtjevne aplikacije.

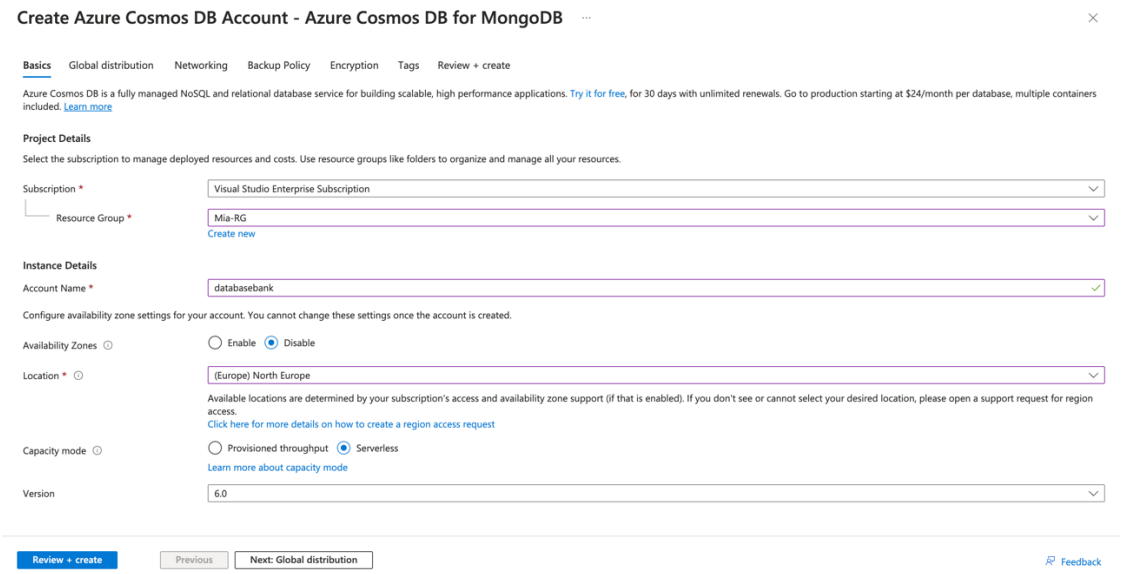
Odabir između ove dvije opcije ovisi o specifičnim zahtjevima aplikacije i preferiranom načinu upravljanja resursima i naplate. U našem slučaju odabrati ćemo Request Units (RU) kao

metriku za upravljanje resursima u Azure Cosmos DB za MongoDB iz razloga što nam pruža fleksibilnost, preciznu naplatu, jednostavnost upravljanja i skalabilnost.

Kada kreiramo novi Azure Cosmos DB račun za MongoDB, možemo konfigurirati sljedeće osnovne postavke računa:

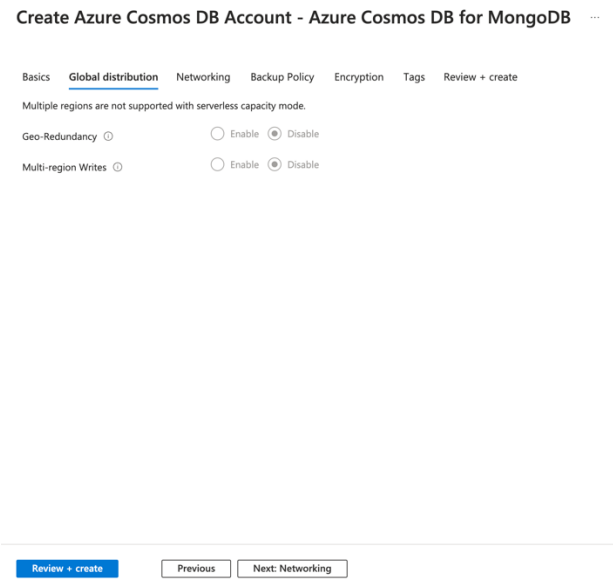
- Pretplata: Ulaznica za korištenje Azure usluga, osigurava pristup resursima i alatima potrebnim za projekte.
- Grupa resursa: Logička jedinica koja grupira povezane resurse na Azure platformi, omogućava organizaciju resursa specifično za račun.
- Naziv računa: Jedinstveni opisni identifikator za vaš Azure Cosmos DB račun, olakšava prepoznavanje među ostalim resursima.
- Lokacija: Azure regija u kojoj želimo smjestiti račun, utječe na brzinu pristupa i latenciju aplikacije.
- Availability Zones: Pružaju višu razinu otpornosti na kvarove, osiguravajući kontinuiran rad aplikacije čak i u slučaju problema u jednoj zoni. U ovoj implementaciji ta opcija je isključena jer ne omogućava takav rad u području zapadne Europe.
- Capacity mode:
    - Provisioned throughput: Unaprijed određujemo resurse za račun, pružajući konzistentan pristup, ali plaćamo fiksnu razinu bez obzira na stvarnu upotrebu.
    - Serverless: Automatski prilagođava resurse stvarnom opterećenju, plaćajući samo za korištene resurse. Idealan za promjenjivo opterećenje ili nepredvidljive zahtjeve, rezultirajući nižim troškovima.

U ovom slučaju odabrana je opcija "Serverless", zato što rezultira nižim troškovima u određenim situacijama, posebno ako aplikacija ima promjenjivo opterećenje ili nepredvidljive zahtjeve. Razlog tome je što plaćamo samo za korištene resurse, što znači da kada aplikacija ima manje posjeta ili aktivnosti, troškovi će biti manji jer ne plaćate za neiskorištene resurse.



Slika 3 Azure Cosmos DB - osnovna konfiguracija

Geografska redundancija omogućava automatsku replikaciju podataka u više regija, osiguravajući visoku dostupnost i otpornost na kvarove. Onemogućavanje ove opcije rezultira pohranom podataka samo u jednoj regiji, što može smanjiti troškove i pojednostaviti upravljanje, ali smanjuje otpornost na kvarove. Pisanje u više regija omogućuje aplikaciji da istovremeno piše podatke u više regija, poboljšavajući performanse i otpornost na kvarove, ali može povećati složenost aplikacije. Odabrana je opcija "disable" za obje komponente jer višestruke regije nisu podržane s serverless kapacitetskim načinom rada.



Slika 4 Globalna distribucija u Azure Cosmos DB-u

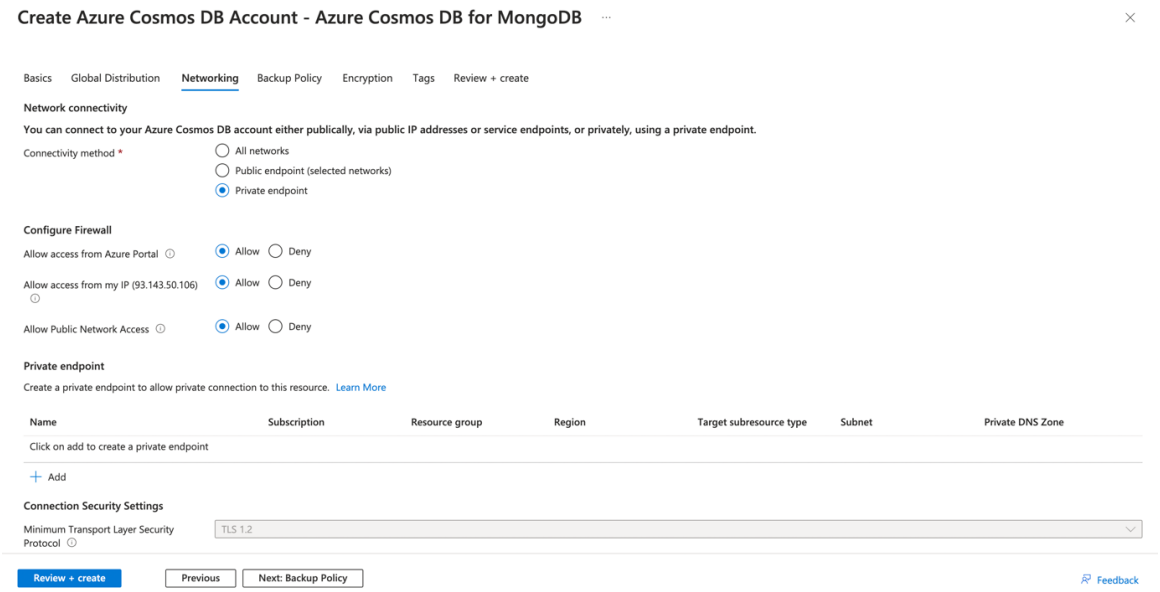
Kada konfiguriramo povezivanje s našim Cosmos DB računom, imamo nekoliko opcija koje trebamo uzeti u obzir:

1. All networks (Sve mreže): Omogućava pristup računu sa svih mreža, što široko otvara račun za pristup iz različitih internetskih mreža.
2. Public endpoint (selected networks): Dopušta pristup računu samo s odabranih mreža putem javnog endpointa, pružajući veću kontrolu nad pristupom.
3. Private endpoint (Privatni endpoint): Najsigurnija opcija, omogućava privatno povezivanje s računom putem privatnog endpointa unutar virtualne mreže, smanjujući površinu napada i osiguravajući sigurniju komunikaciju.

U kontekstu konfiguracije vatrozida, imamo sljedeće opcije:

1. Allow access from Azure Portal: Omogućava pristup računu iz Azure portala radi administrativnih zadataka.
2. Allow access from my IP (93.143.50.106): Ograničava pristup samo s određene IP adrese, korisno za sigurnosno ograničavanje pristupa samo na odabrane lokacije.
3. Allow Public Network Access: Omogućava pristup računu izvan virtualne mreže, što može biti potrebno za određene vrste aplikacija.

U ovom slučaju, zbog visokih sigurnosnih zahtjeva, odabrana je opcija Private endpoint i konfiguriran vatrozid prema specifičnim potrebama aplikacije kako bi se osigurala maksimalna sigurnost i privatnost podataka.

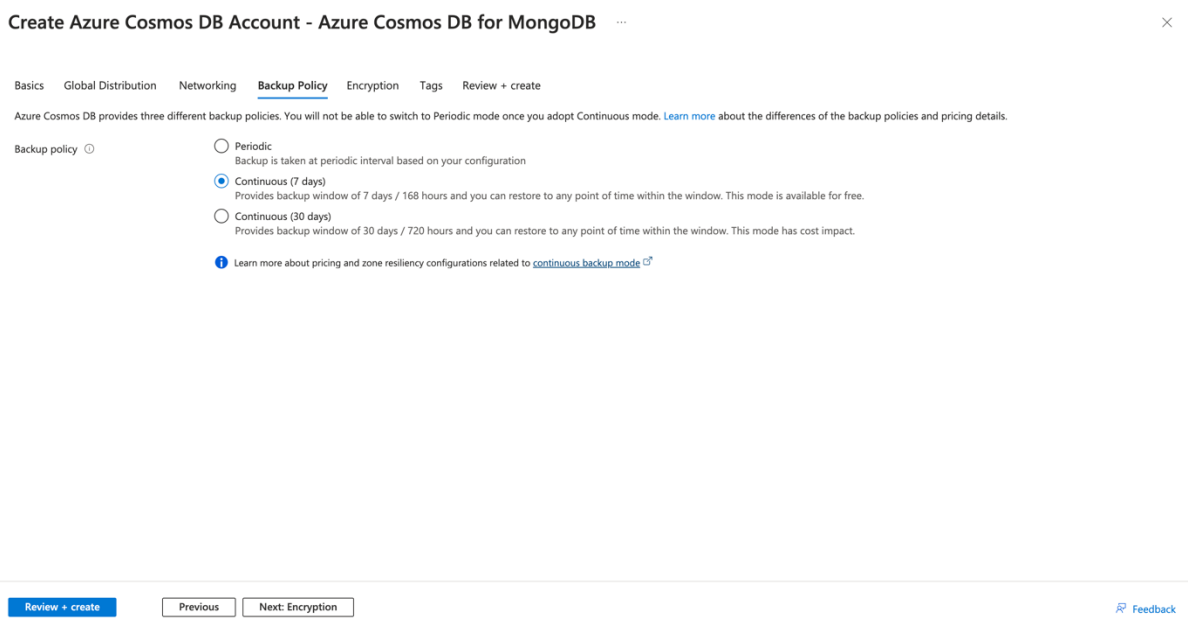


Slika 5 Azure Cosmos DB - Mrežno povezivanje

Azure Cosmos DB nudi tri politike sigurnosnih kopija za vaše podatke:

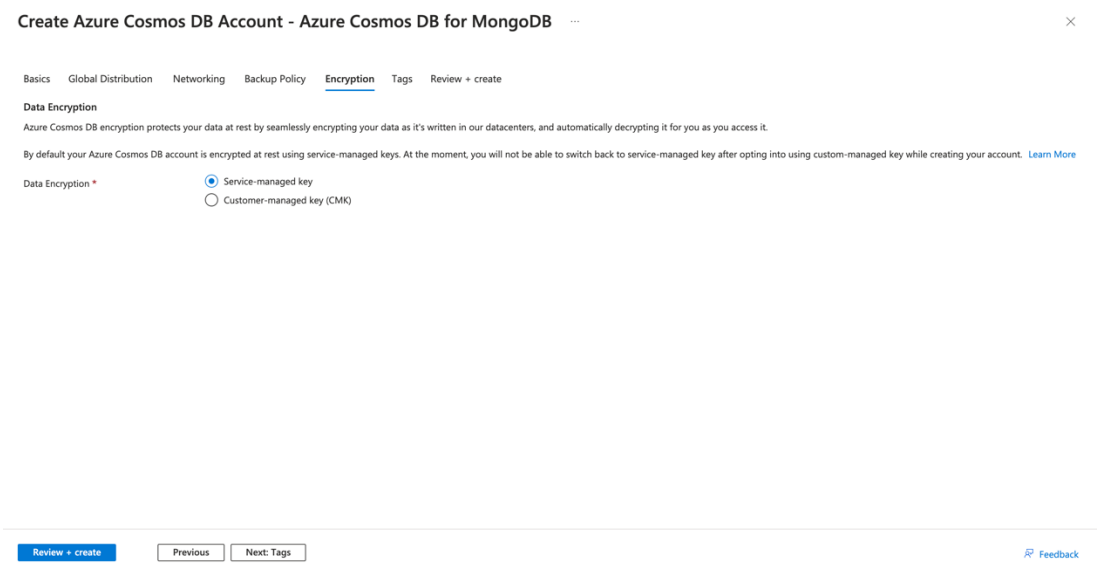
1. Periodic (Periodički): Sigurnosna kopija se uzima u periodičkim intervalima prema konfiguraciji.
2. Continuous (7 dana): Pruža prozor sigurnosne kopije od 7 dana ili 168 sati te omogućava vraćanje baze podataka u bilo koju točku unutar tog prozora, također je besplatan.
3. Continuous (30 dana): Pruža prozor sigurnosne kopije od 30 dana ili 720 sati te ima mogućnost vraćanja baze podataka u bilo koju točku unutar tog prozora.

Odabrana je politika Continuous (7 dana) za sigurnosnu kopiju jer pruža dovoljno vremena za obnovu podataka u slučaju kvarova i besplatna je, što je prikladno za implementiranu aplikaciju.



Slika 6 Azure Cosmos DB postavljanje sigurnosne kopiije

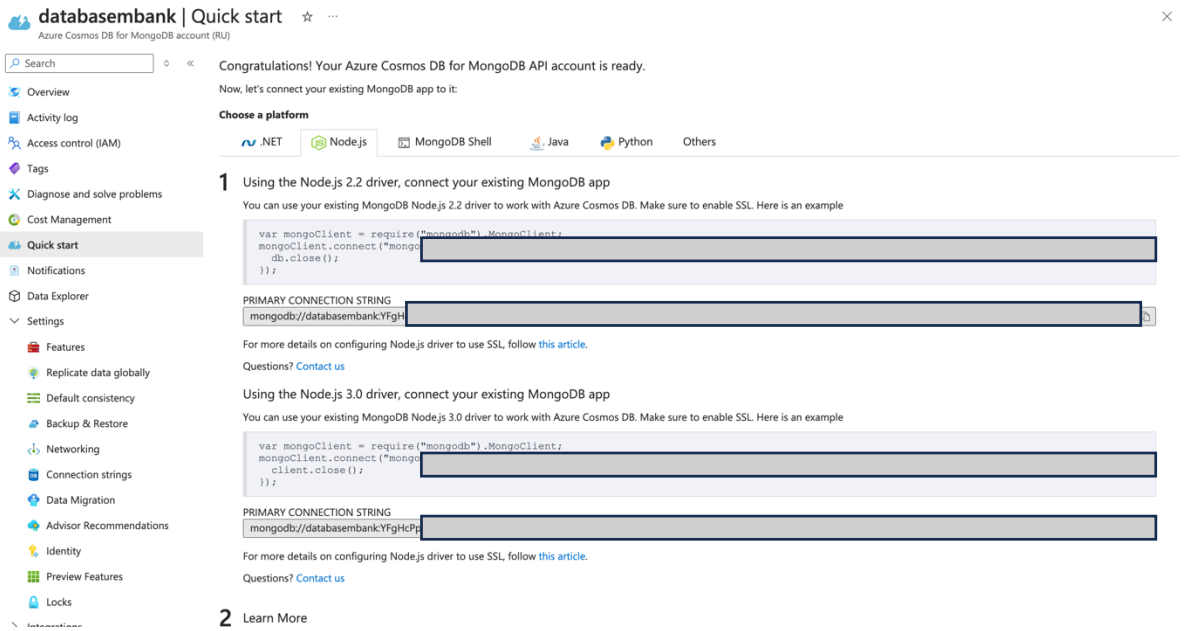
Enkripcija podataka u Azure Cosmos DB štiti informacije u mirovanju tako što ih automatski šifrira dok se pohranjuju u podatkovne centre te ih po potrebi dešifrira kada im pristupamo. Osnovna postavka zaštite podataka podrazumijeva korištenje ključeva upravljanih uslugom, što osigurava sigurnost podataka.



Slika 7 Azure Cosmos DB enkripcija podataka

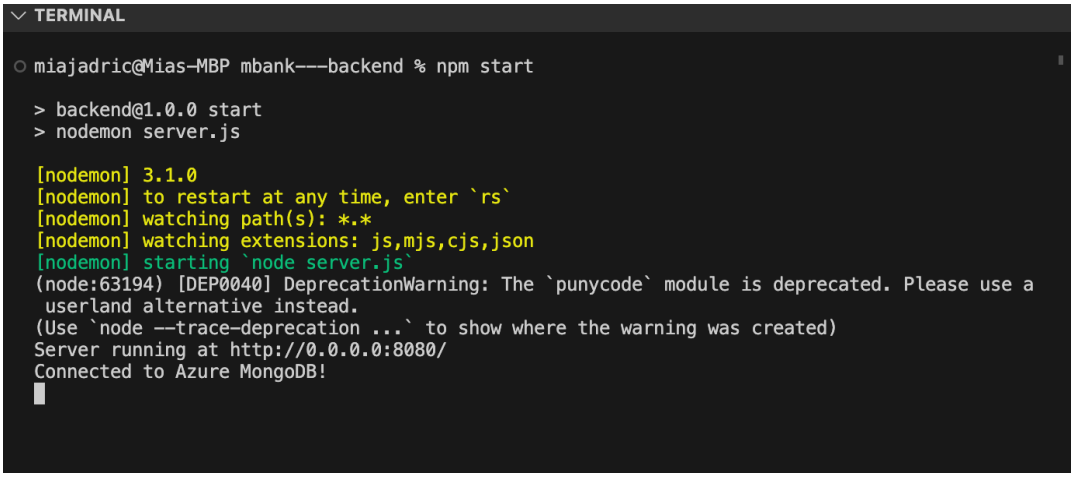
4.1.2 Povezivanje s Azure Cosmos DB-om za MongoDB

Kada se povezujemo s Azure Cosmos DB, prvo što radimo je odabir odgovarajućeg connection stringa iz driver-a, koji je kompatibilan s našom aplikacijom. Zatim, taj connection string dodajemo u našu .env datoteku kako bismo ga mogli koristiti unutar naše aplikacije. To je kao ključ koji nam omogućava da se povežemo s bazom podataka.



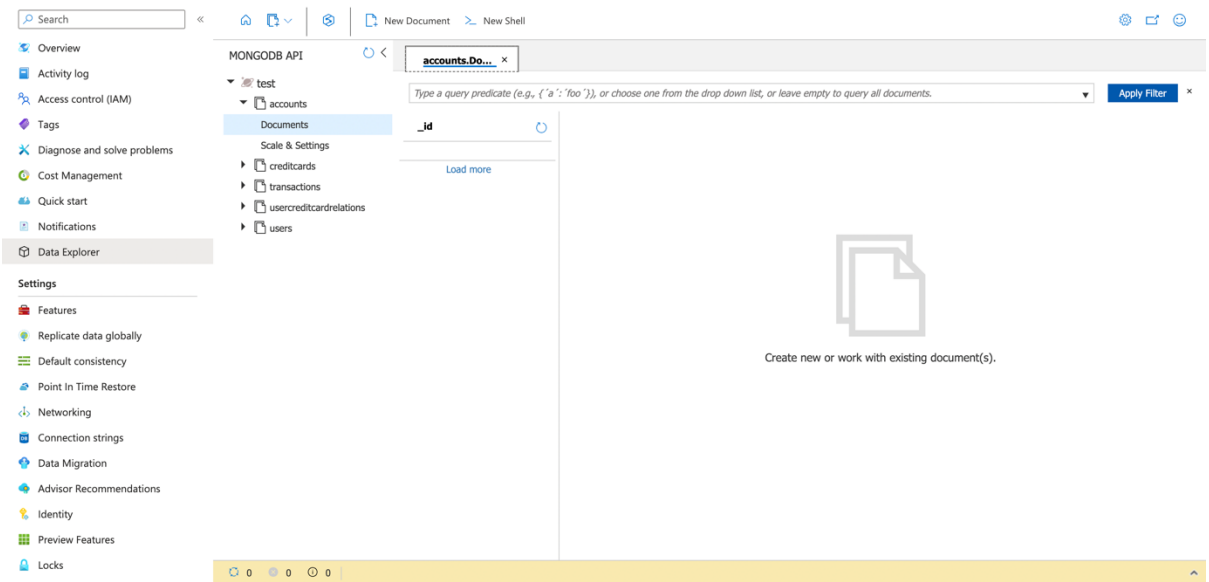
Slika 8 Azure Cosmos DB - connection string

Nakon što smo to učinili, pokrećemo naš backend. Kada se backend pokrene, on pokušava uspostaviti vezu s bazom podataka koristeći taj connection string. Ukoliko je povezivanje uspješno, vidjet ćemo poruku u konzoli koja potvrđuje uspješnu vezu.



Slika 9 Lokalni server povezan s Azure Cosmos DB

Zatim otvaramo Azure portal kako bismo provjerili jesu li dokumenti pravilno spremljeni u bazu podataka. To radimo pomoću alata kao što je Data Explorer, gdje možemo vidjeti stvorene dokumente i provjeriti jesu li oni onakvi kakve smo očekivali.



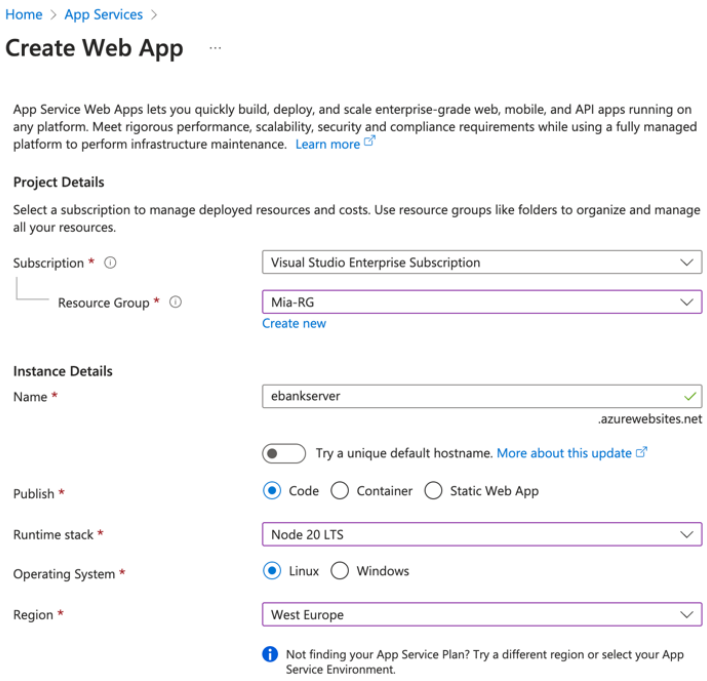
Slika 10 Azure Cosmos DB – preglednik podataka

4.2 Implementacija Azure App Service za Express.js backend

App Service Web Apps omogućava brzu izgradnju, implementaciju i skaliranje web, mobilnih i API aplikacija visoke klase koje se izvode na bilo kojoj platformi. Pruža sve što vam je potrebno za zahtjeve za performansama, skalabilnošću, sigurnošću i usklađenošću, koristeći potpuno upravljenu platformu za održavanje infrastrukture.

U prvom koraku, korisnik postavlja osnovne informacije potrebne za stvaranje web Web App-a:

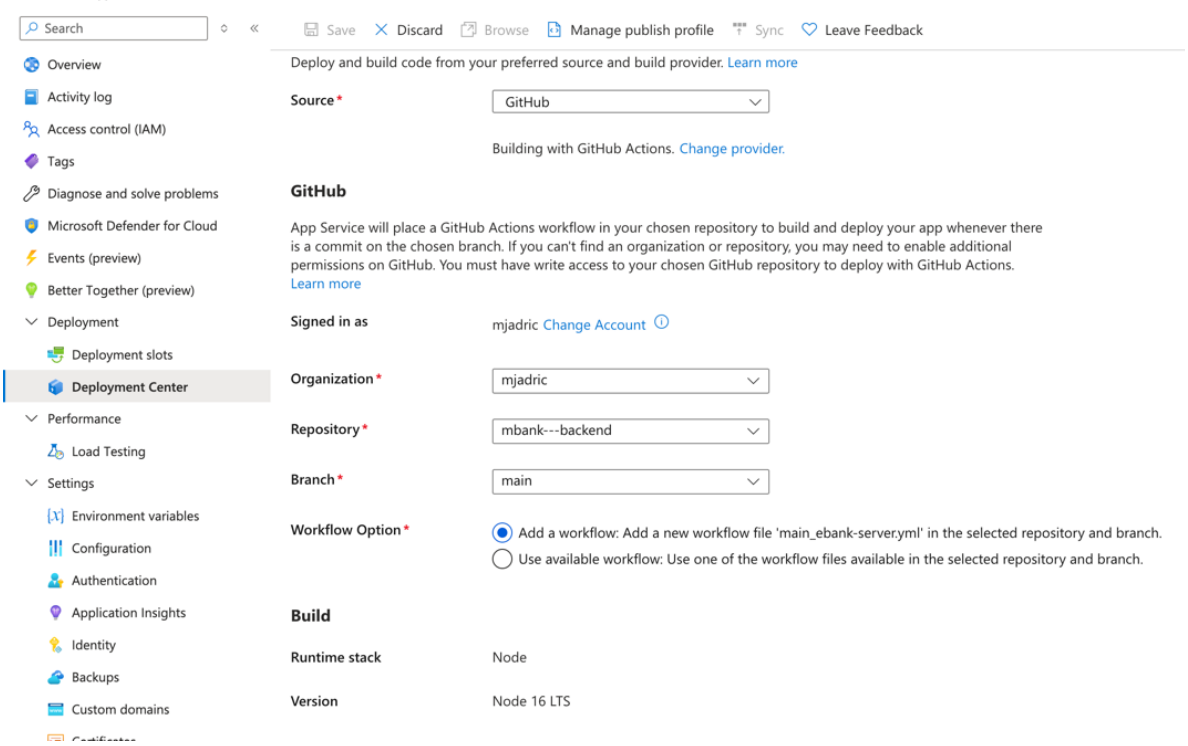
- Naziv web aplikacije: Jedinstveno ime koje identificira vašu web aplikaciju.
- Pretplata: Odabir pretplate koja će pokrivati troškove korištenja web aplikacije.
- Resurs grupa: Odabir ili stvaranje grupe resursa u kojoj će se smjestiti web aplikacija.
- Izvršni stog: Odabir tehnologije ili platforme za izvršavanje web aplikacije, poput .NET, Node.js, Python itd.
- Operacijski sustav: Odabir operativnog sustava na kojem će se izvršavati web aplikacija, kao što su Windows ili Linux.
- Regija: Odabir regije unutar odabrane lokacije za hostiranje web aplikacije.



Slika 11 Azure Web App - osnovna konfiguracija

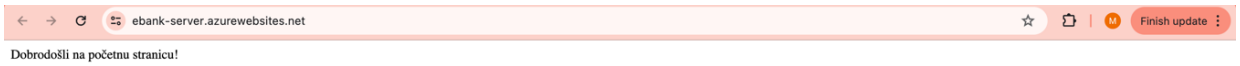
U drugom koraku, nakon što smo postavili osnovne informacije u Azure Portalu za stvaranje Web App-a, prelazimo na deployment backenda s GitHuba. U procesu konfiguracije deploymenta, prvo pristupamo postavkama Web App-a u Azure Portalu. Tamo tražimo opciju za deployment iz GitHuba. Kada pronađemo tu opciju, povezujemo svoj GitHub račun s Azureom i odaberemo željeni repozitorij i granu za deployment.

Nakon što smo postavili deployment, slijedi konfiguracija build procesa. Ovdje postavljamo proces koji automatski preuzima kod s odabrane grane na GitHubu i izvršava build.



Slika 12 Deployment backenda

Kada je deployment završen, Azure automatski generira URL adresu za App Service putem kojeg šaljemo zahtjeve serveru. Ova URL adresa omogućava pristup aplikaciji preko interneta, čime korisnici mogu interagirati s web servisima.



Slika 13 Uspješno pokrenut Azure Web App

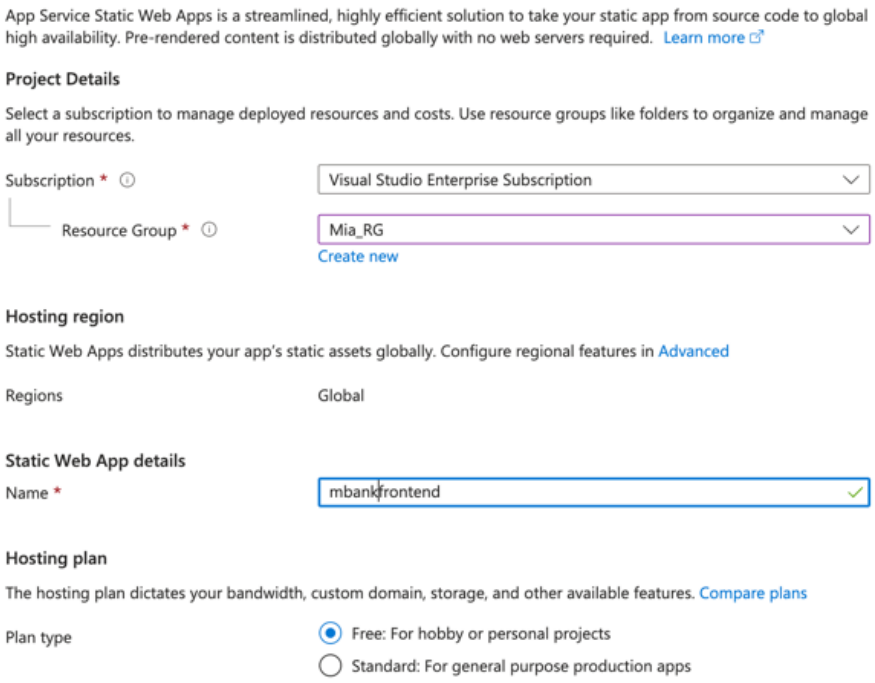
4.3 Implementacija Static Web App za React.js frontend

Azure Static Web Apps omogućava brzu i jednostavnu implementaciju React.js aplikacija na Azure platformi, uz automatsko stvaranje build i implementacijskog pipelinea putem GitHub Actions. Osim toga, pruža integraciju s drugim Azure uslugama, što omogućava izgradnju full-stack web aplikacija. Osigurava visoke performanse i dostupnost, uz automatsko skaliranje resursa prema potrebama aplikacije.

4.3.1 Stvaranje Azure Static Web App

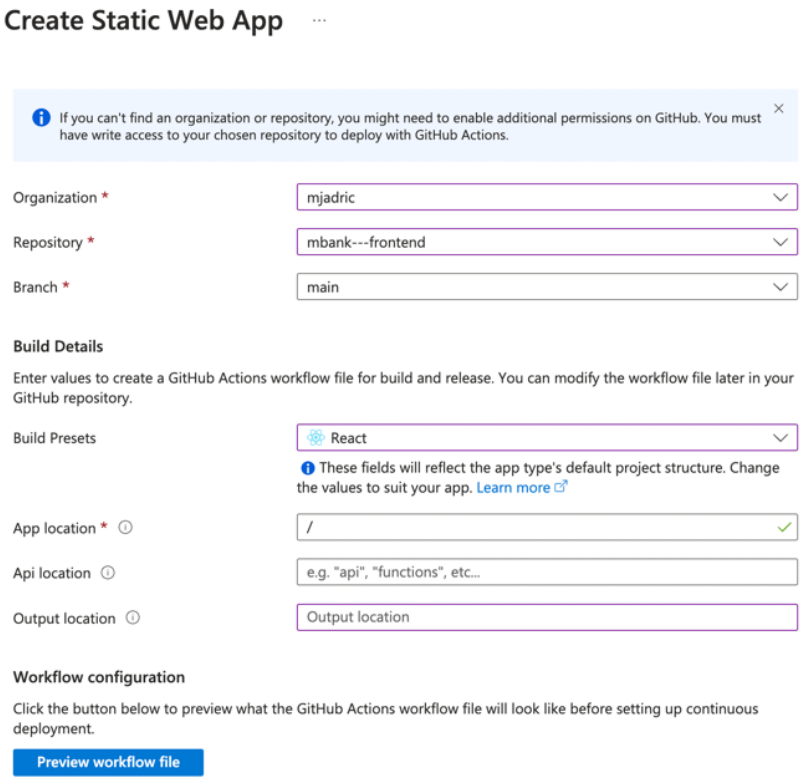
U prvom koraku korisnik postavlja osnovne informacije potrebne za stvaranje Static Web App-a:

- Naziv aplikacije: Korisnik određuje naziv aplikacije koji će identificirati njegovu web aplikaciju u Azure okruženju.
- Hosting plan: Korisnik odabire između besplatnog i standardnog plana hostinga. Besplatni plan pogodan je za hobi ili osobne projekte, dok se standardni plan preporučuje za produkcijske aplikacije.



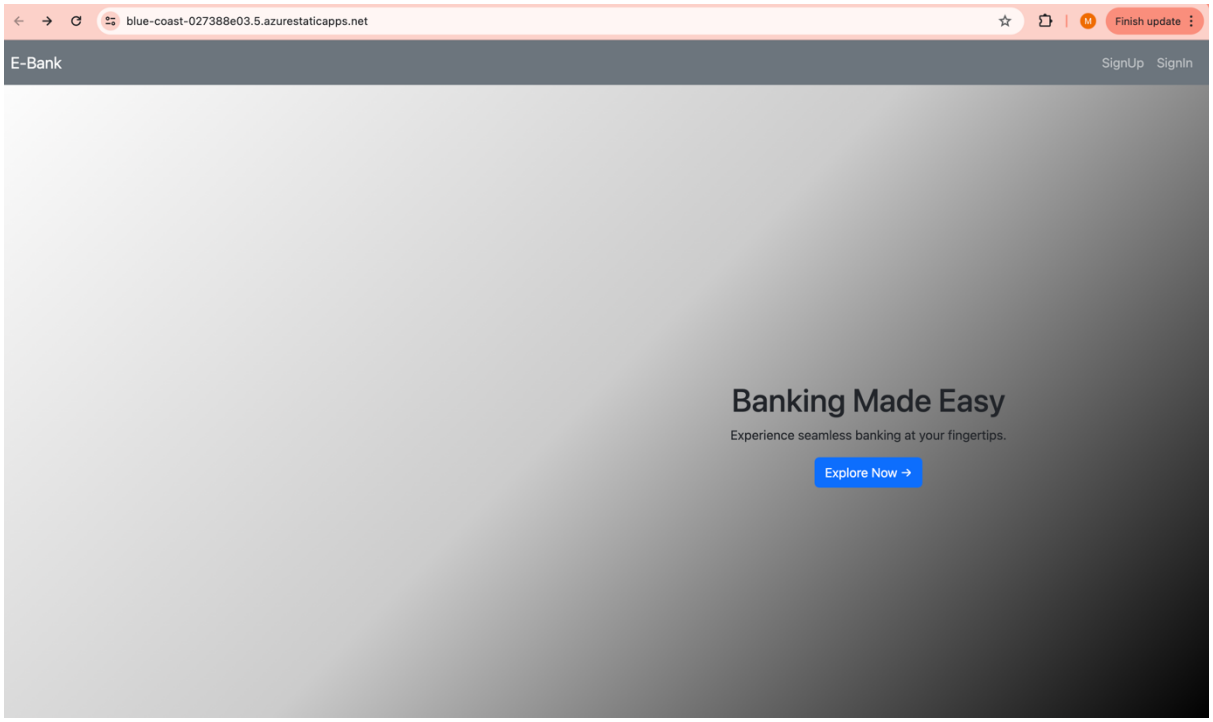
Slika 14 Azure Static Web App - osnovna konfiguracija

- Izvor implementacije: Korisnik odabire izvor implementacije, što može biti GitHub, Azure DevOps ili drugi repozitorij. U ovom slučaju odabran je GitHub, zbog svoje kontinuiranosti tijekom deployanja aplikacije.
- Informacije o GitHub računu: Korisnik unosi informacije o svom GitHub računu, uključujući ime korisnika ili organizacije te naziv repozitorija. Ovo omogućava Azure-u pristup izvornom kodu aplikacije radi automatske implementacije i upravljanja putem GitHub akcija.



Slika 15 Azure Static Web App - povezivanje s GitHubom

Kada je deployment završen, Azure automatski generira domenu za frontend aplikaciju putem koje možemo pristupiti web aplikaciji. Ova domena omogućava korisnicima da jednostavno pristupe aplikaciji putem internetskog preglednika bez potrebe za ručnim konfiguriranjem ili postavljanjem DNS zapisa. To olakšava distribuciju aplikacija i omogućava korisnicima brz i jednostavan pristup novim verzijama web aplikacija nakon implementacije.



Slika 16 Azure Static Web App pokrenut u browseru

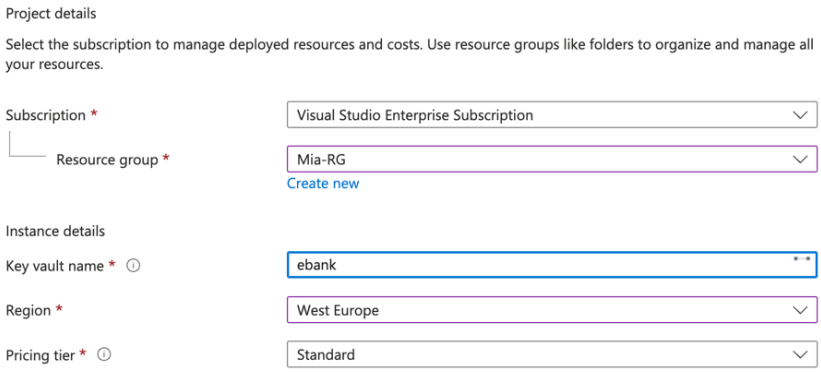
5 Implementacija sigurnosnih mjera u Azure portalu

U ovom poglavlju detaljno ćemo opisati sigurnosne mjere implementirane za zaštitu aplikacije hostane na Azure platformi.

5.1 Implementacija Azure Key Vault-a

Azure Key Vault je usluga upravljanja tajnama, ključevima i certifikatima u Azure oblaku. Implementacija Azure Key Vaulta omogućuje sigurno pohranjivanje i upravljanje osjetljivim podacima, poput connection stringova, certifikata i tajni.

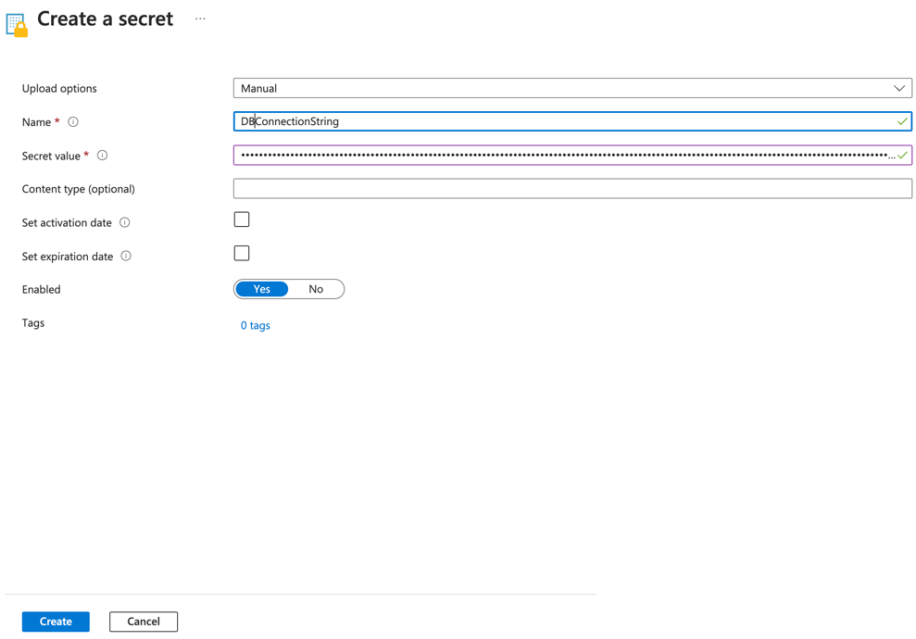
Prvi korak u implementaciji je kreiranje Azure Key Vaulta, gdje se odabiru osnovne postavke poput resurs grupe, naziv, regija te postojeće resursne grupe kao i u prethodnim koracima.



Slika 17 Azure Key Vault - osnovna konfiguracija

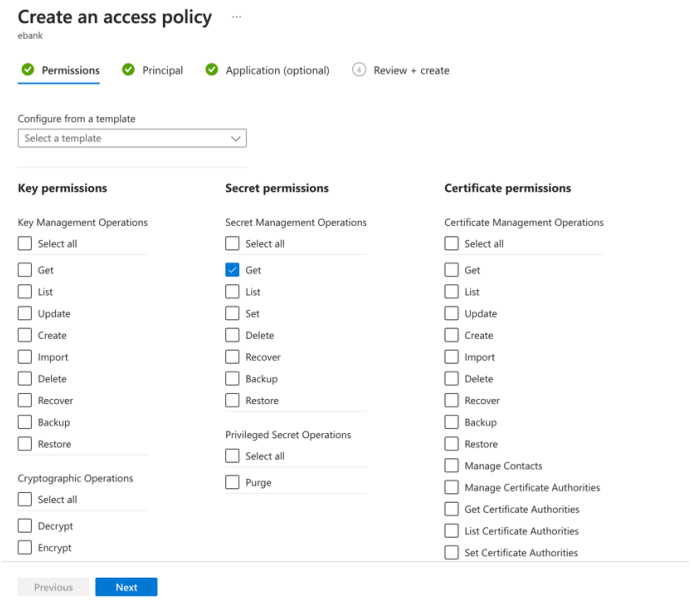
Nakon što je Azure Key Vault kreiran, možemo dodati tajne koje želimo pohraniti. Prilikom dodavanja nove tajne, prvo unosimo naziv tajne. Naziv tajne treba biti jedinstven unutar Key Vaulta i jasno identificirati tajnu koju pohranjujemo. U ovom slučaju, naziv tajne bi mogao biti "DBConnectionString". Nakon unosa naziva, slijedi unos vrijednosti tajne. Vrijednost tajne predstavlja osjetljive podatke koje želimo pohraniti, poput connection stringa, lozinke ili certifikata. U ovom primjeru, vrijednost tajne bio bi MongoDB connection string koji omogućuje povezivanje s Azure Cosmos DB bazom podataka.

Dodatno, možemo konfigurirati dodatne postavke kao što je postavljanje vremena isteka tajne. Ovo omogućuje kontrolu nad trajanjem tajne i automatsko brisanje ili deaktiviranje tajne nakon isteka određenog vremenskog perioda. Ova postavka je korisna za osiguravanje sigurnosti osjetljivih podataka i sprječavanje njihovog neovlaštenog pristupa nakon isteka roka važenja. Međutim, u ovom slučaju to nije bilo potrebno jer je odabrano da tajna bude trajna, kako bi se osigurao kontinuirani pristup bazi podataka iz aplikacije.



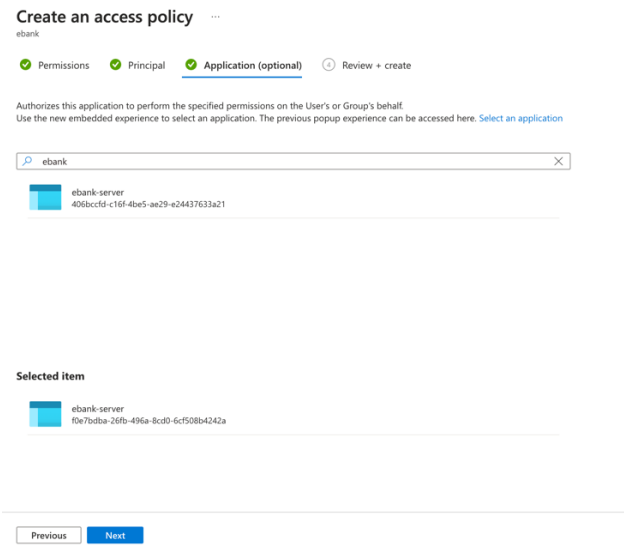
Slika 18 Dodavanje tajni u Azure Key Vault

Kada je tajna pohranjena u Azure Key Vault, potrebno je konfigurirati pristup aplikaciji kako bi mogla dohvatiti tu tajnu. Prvi korak u ovom procesu je dodijeliti pristup aplikaciji na Azure Key Vaultu korištenjem Identity and Access Management (IAM). To se može postići dodjeljivanjem uloge s pravom pristupa tajni "Get" ili prilagođavanjem uloga prema specifičnim potrebama.



Slika 19 Dodjeljivanje pristupa tajni u Azure Key Vault-u

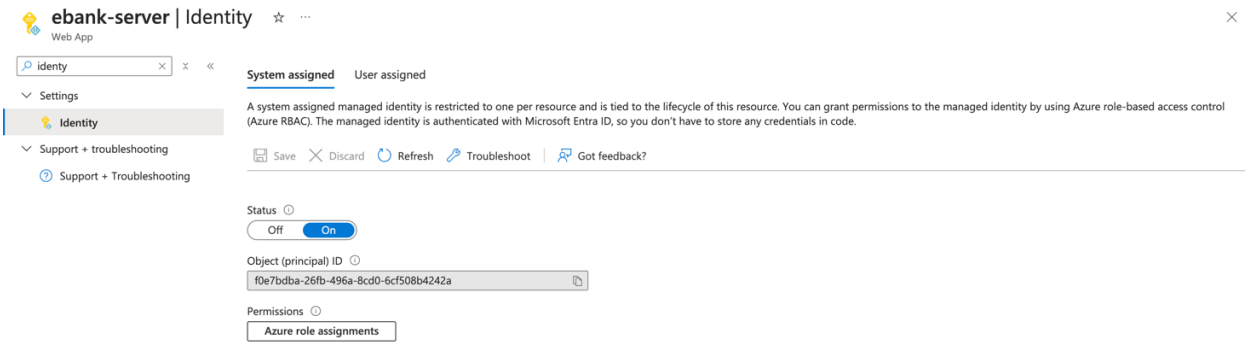
Kada dodjeljujemo pristup tajnama u Key Vaultu, također biramo principal, odnosno korisnički račun (u ovom slučaju je odabran vlastiti korisnički račun) te biramo aplikaciju koja će imati pristup određenoj tajni (u ovom slučaju ebank-server). Ovo omogućava precizno upravljanje pristupom tajnama unutar Key Vaulta i osigurava da samo ovlaštene aplikacije i korisnici mogu dohvatiti osjetljive informacije.



Slika 20 Odabir aplikacije koja ima pristup tajni

Sljedeći korak je omogućiti sustavski dodijeljenu upravljivu identitetu (system assigned managed identity) za web aplikaciju. Ovaj identitet je vezan uz životni ciklus resursa i omogućava dodjeljivanje ovlasti korištenjem Azure uloga temeljenih na pristupu (Azure RBAC). Autentificiran putem Microsoftova unosa ID-a, što znači da nije potrebno pohranjivati nikakve vjerodajnice u kodu aplikacije. Ovo osigurava siguran i praktičan način za upravljanje pristupom Azure resursima iz web aplikacije.

I



Slika 21 Identitet upravljan od strane sustava s dodijeljenom ulogom

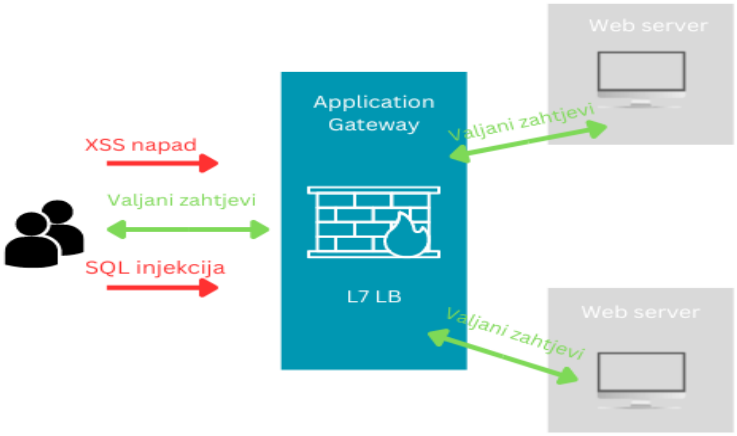
Da bismo dohvatili connection string iz Azure Key Vault-a i koristili ga u aplikaciji, prvi korak je instalacija paketa potrebnih za rad s Azure Key Vaultom. To možemo postići pokretanjem naredbe `npm install @azure/keyvault-secrets @azure/identity`. Ovi paketi omogućavaju komunikaciju s Azure Key Vaultom te dohvat tajni.

Slika 22 Kod za dohvaćanje tajni

U ovom kodu, prvo uvozimo potrebne module iz paketa `@azure/identity` i `@azure/keyvault-secrets`. Zatim postavljamo varijable koje sadrže informacije o Azure Key Vaultu i konfiguriramo `SecretClient` objekt koji će se koristiti za interakciju s Key Vaultom. Ključna stvar u ovom dijelu koda je korištenje `DefaultAzureCredential` za stvaranje identiteta koji će se koristiti za autentifikaciju pri pristupu Azure Key Vaultu. Ovaj identitet automatski koristi prijavu sustava ili identitet servisa pridružen Azure okruženju u kojem se aplikacija izvršava, što znači da nema potrebe za eksplicitnim upravljanjem vjerodajnicama. Zatim definiramo funkciju `getSecret(secretName)` koja dohvaća tajnu iz Key Vaulta na temelju imena tajne. Ova funkcija koristi `client.getSecret(secretName)` za dohvaćanje tajne, a zatim vraća vrijednost tajne. U funkciji `startServer()`, koristimo `getSecret()` za dohvaćanje connection stringa za MongoDB iz Key Vaulta pomoću imena tajne `MONGO_URI_SECRET_NAME`. Nakon toga, koristimo dobiveni connection string za uspostavljanje veze s MongoDB bazom podataka pomoću biblioteke `mongoose`. Nakon uspješne veze, ispisujemo poruku "Connected to Azure MongoDB!".

5.2 Azure Application Gateway-a i Web Application Firewall-a

Azure Application Gateway je usluga za isporuku aplikacija koja omogućuje upravljanje prometom na web aplikacijama. Pruža balansiranje opterećenja na sedmom sloju (L7) OSI modela, što znači da može donositi odluke o usmjeravanju prometa na temelju sadržaja aplikacijskog sloja, kao što su URL putanje i sadržaj zaglavlja. Web Application Firewall je sastavni dio Azure Application Gatewaya koji pomaže u zaštiti web aplikacija od uobičajenih napada poput SQL injekcija, skriptiranja između stranica (XSS) i drugih prijetnji iz OWASP Top 10.



Slika 23 Arhitektura mrežnog prometa s Application Gateway i Web Application Firewall zaštitom

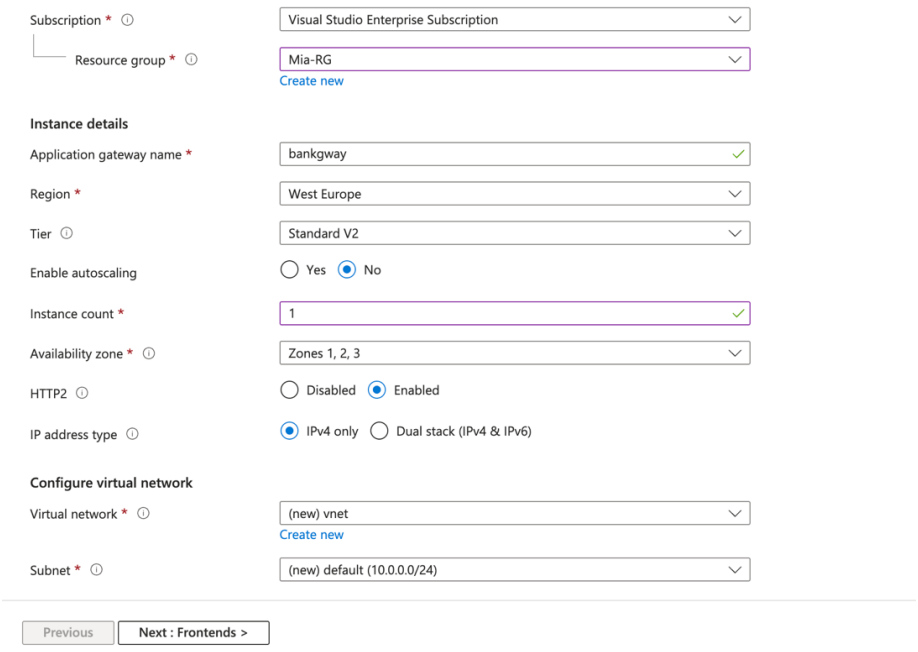
Na slici se vidi kako svi zahtjevi prema web aplikacijama prvo prolaze kroz Azure Application Gateway, gdje WAF analizira ove zahtjeve. Maliciozni zahtjevi poput XSS napada i SQL injekcija se detektiraju i blokiraju, dok se validni zahtjevi propuštaju i dalje distribuiraju prema odredišnim web aplikacijama smještenim na različitim mjestima. Azure Application Gateway također vrši balansiranje opterećenja, distribuirajući zahtjeve ravnomjerno između poslužitelja prema unaprijed definiranim pravilima.

Korištenje Azure Application Gatewaya s WAF-om donosi brojne prednosti, uključujući poboljšanu sigurnost kroz efikasno blokiranje malicioznih napada prije nego što dođu do web aplikacija, ravnomjernu distribuciju prometa za bolje performanse i pouzdanost aplikacija, centralizirano upravljanje i kontrolu pristupa te jednostavno skaliranje resursa prema

potrebama aplikacije. Ova arhitektura osigurava sveobuhvatnu zaštitu i efikasno upravljanje prometom, čineći web aplikacije sigurnijima i pouzdanijima.

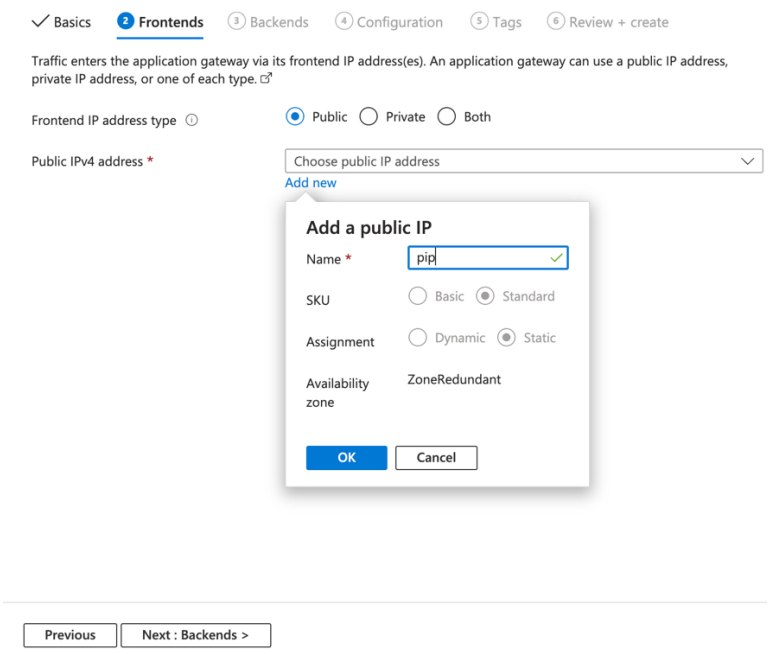
5.2.1 Implementacija Azure Application Gateway-a

Implementacija Azure Application Gateway-a započinje odabirom pretplate i resursne grupe. Pretplata "Visual Studio Enterprise Subscription" omogućuje pristup potrebnim resursima, dok resursna grupa "Mia-RG" služi kao kontejner za sve povezane resurse. Uneseni su osnovni detalji o instanci: naziv Application Gateway-a "bankgway" radi lakše identifikacije, regija "West Europe" zbog smanjenja latencije i poštivanja zakonskih regulativa, te tier "Standard V2" zbog poboljšanih performansi i značajki. Automatsko skaliranje postavljeno je na "No" kako bi se ručno upravljalo brojem instanci, a broj instanci je postavljen na 1 za početak. Odabrane su zone dostupnosti (Zones 1, 2, 3) radi veće otpornosti na kvarove i bolje dostupnosti, jer raspodjeljuju resurse na različite fizičke lokacije unutar iste regije. HTTP2 je omogućen za poboljšanje učinkovitosti prijenosa podataka i smanjenje latencije. Tip IP adrese postavljen je na "IPv4 only" jer većina aplikacija i dalje koristi IPv4 adrese. Kreirana je nova virtualna mreža (vnet) i podmreža (subnet) s nazivom "default (10.0.0.0/24)" kako bi se omogućila mrežna komunikacija između resursa.



Slika 24 Application Gateway - osnovna konfiguracija

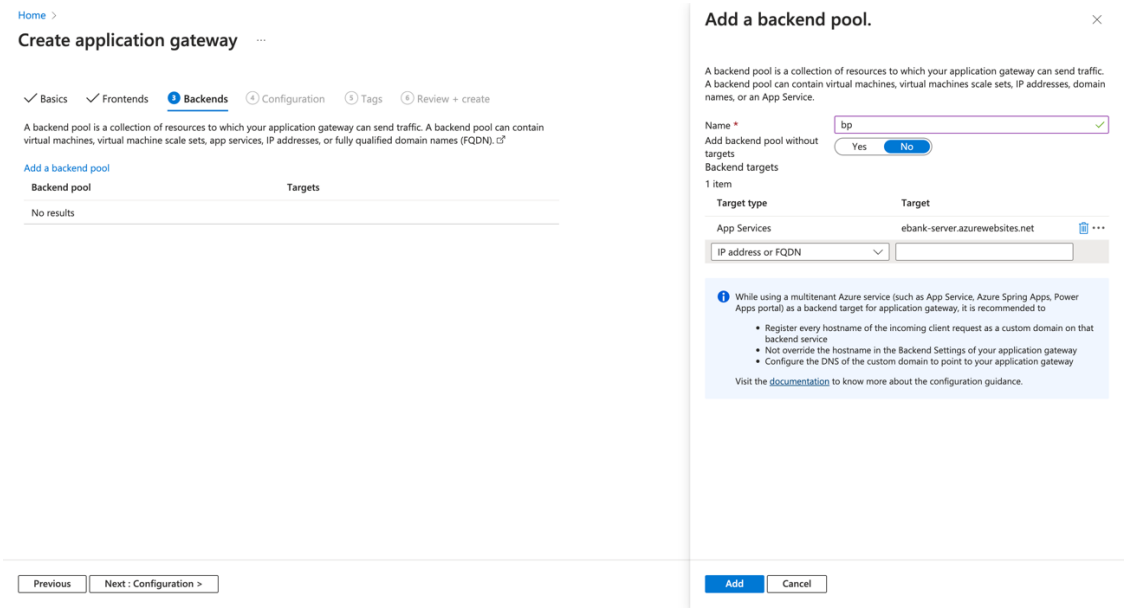
U sljedećem koraku konfigurirane su frontend postavke. Tip IP adrese za frontend postavljen je na "Public" kako bi aplikacija bila dostupna putem interneta. Kreirana je nova javna IP adresa s nazivom "pip1", SKU "Standard" zbog bolje performanse i skalabilnosti, dodjela je postavljena na "Static" kako bi IP adresa ostala stalna, te zona dostupnosti na "ZoneRedundant" kako bi se osigurala otpornost na kvarove.



Slika 25 Application Gateway postavljanje frontend IP adrese

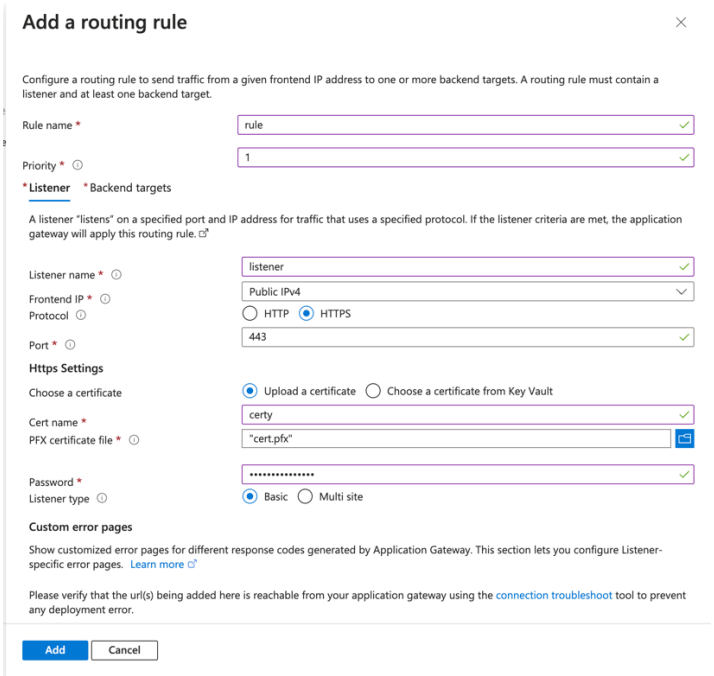
Nakon konfiguriranja frontend postavki, slijedi konfiguriranje backend postavki koje će omogućiti komunikaciju između Application Gateway-a i backend resursa.

Otvoren je prozor za dodavanje novog backend pool-a, koji je nazvan "bp". Odabrana je opcija za dodavanje backend pool-a bez ciljeva, što omogućava naknadno dodavanje ciljeva prema potrebi. U polju "Backend targets" dodan je App Service "ebank-server.azurewebsites.net", što omogućava Application Gateway-u usmjeravanje prometa prema ovom resursu.



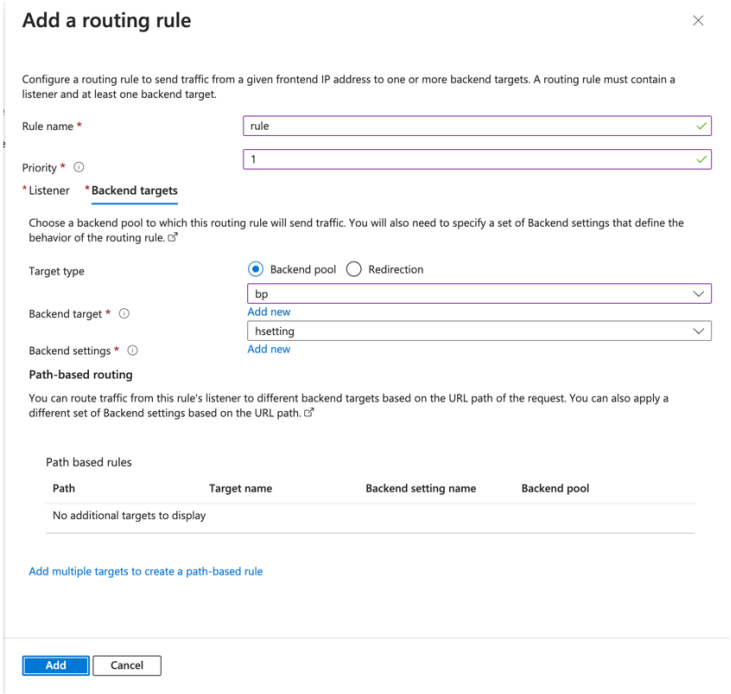
Slika 26 Application Gateway postavljanje backend pool-a

Kreirano je novo routing pravilo nazvano "rule" s prioritetom "1", što znači da će ovo pravilo imati najviši prioritet prilikom usmjeravanja prometa. U sekciji "Listener" konfiguriran je novi listener nazvan "listener", koji koristi protokol "HTTPS" na portu "443". Listener je vezan na prethodno konfiguriranu javnu IP adresu (Public IPv4). Za navedeni protokol potreban je i certifikat koji je generiran i autoriziran od strane Azure Key Vault-a.



Slika 27 Application Gateway postavljanje listener-a

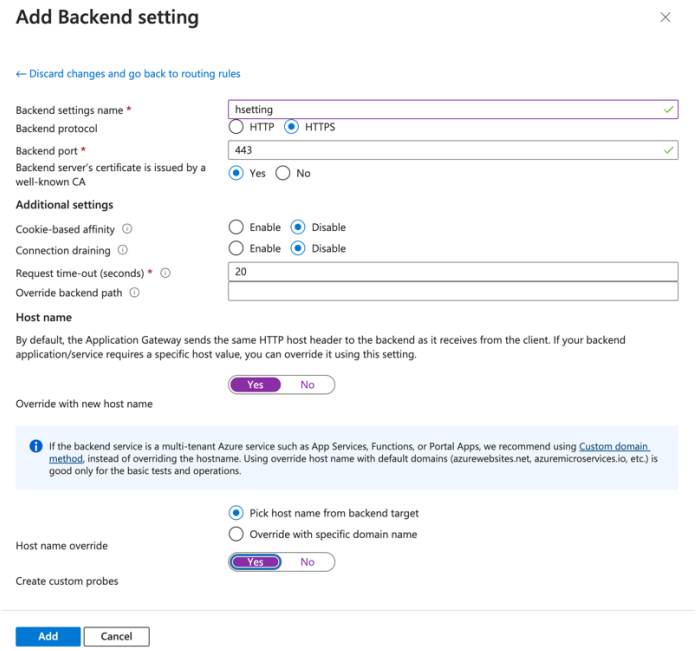
Sljedeći korak je konfiguracija pravila usmjeravanja kako bi se promet s frontend IP adrese usmjerio na odgovarajući backend pool. Kreiranje pravila usmjeravanja uključuje definiranje naziva pravila, postavljanje prioriteta, odabir listenera, te odabir backend targeta i postavki. U ovom slučaju, pravilo je nazvano "rule", prioritet je postavljen na 1, a backend target je "bp" s postavkama "hsetting". Path-based routing također se može koristiti za definiranje specifičnih putanja kako bi se promet usmjeravao prema različitim backend pool-ovima bazirano na URL putanji zahtjeva.



Slika 28 Application Gateway postavljanje backend target-a

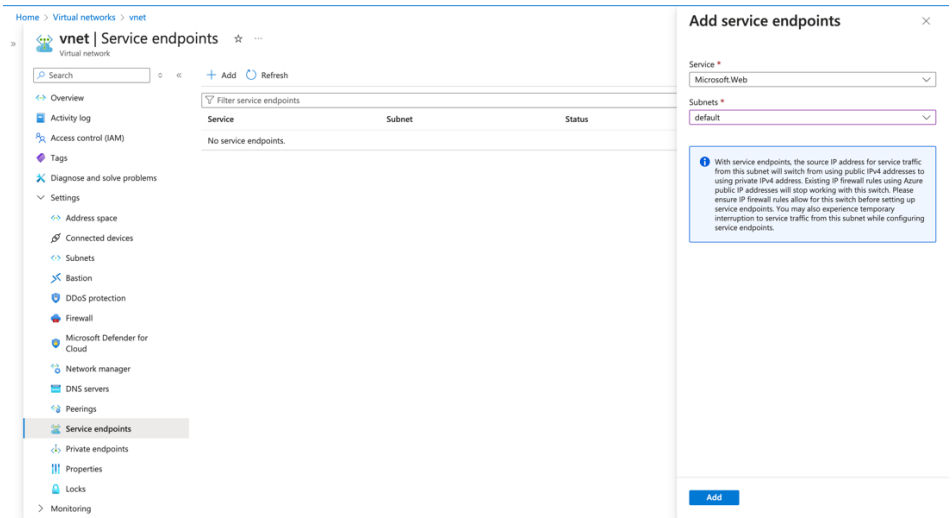
Kreirane su backend postavke nazvane "hsetting" s protokolom "HTTPS" i portom "443". Dodatne postavke uključuju: Cookie-based affinity, koja je onemogućena, što znači da će svaki novi zahtjev korisnika biti tretiran kao zaseban i neće biti povezan s prethodnim zahtjevima putem kolačića. Connection draining također je onemogućen, što znači da trenutne konekcije neće biti završene prije nego što backend resurs postane nedostupan, što može dovesti do gubitka podataka ili prekida usluge. Request time-out postavljen je na 20 sekundi, što definira maksimalno vrijeme čekanja na odgovor od backend resursa. Omogućena je opcija "Override backend path", koja omogućava definiranje specifičnih putanja za backend resurse. Također je omogućena opcija "Host name override" s odabirom "Override with specific domain name",

što omogućava da Application Gateway šalje specifični host header prema backendu, što je korisno ako backend aplikacija zahtijeva određeni host header za ispravno funkcioniranje.



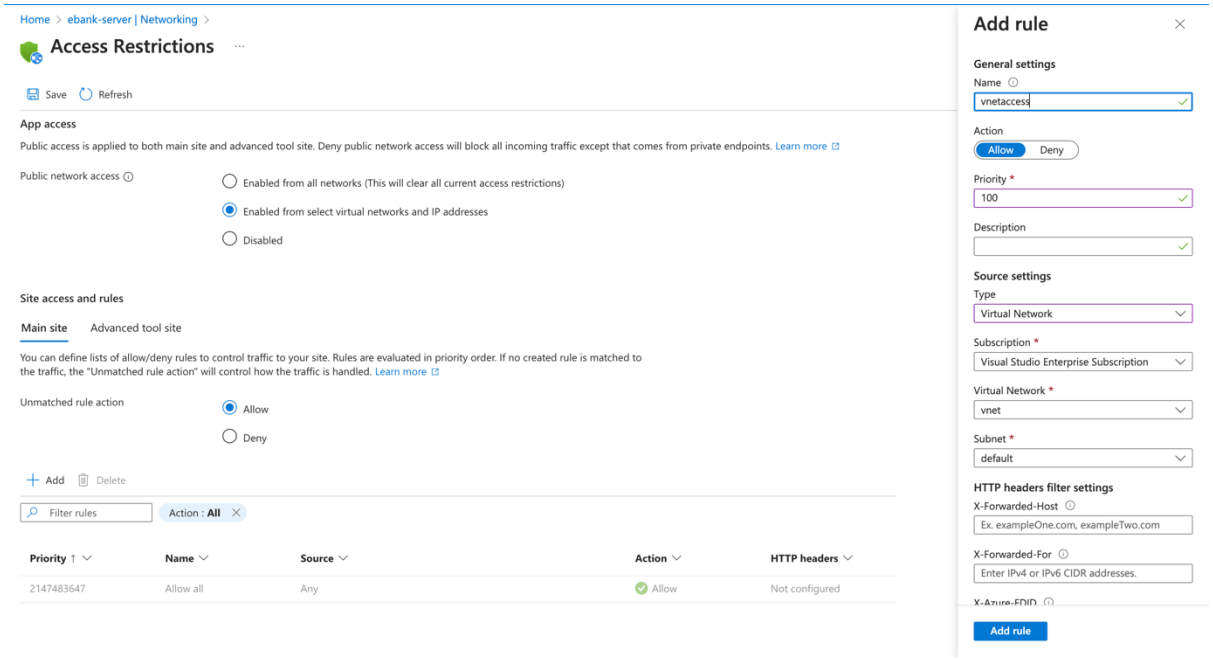
Slika 29 Application Gateway postavljanje backend postavki

Dodavanje uslužnih krajnjih točaka (Service Endpoints) u virtualnu mrežu (VNet) omogućava korištenje privatnih IP adresa umjesto javnih za promet prema odabranom servisu. U ovom slučaju, servis je "Microsoft.Web" a podmreža je "default". Ova konfiguracija pomaže u povećanju sigurnosti i performansi aplikacija.



Slika 30 Postavljanje uslužnog kraja virtualne mreže

Postavljanje pravila pristupa omogućava kontrolu pristupa aplikaciji putem javne mreže. U postavkama restrikcija pristupa, javni mrežni pristup je omogućen sa svih mreža. Dodano je pravilo pristupa nazvano "vnetaccess" koje omogućava pristup iz virtualne mreže "vnet" unutar podmreže "default". Akcija je postavljena na "Allow" s prioritetom 100.

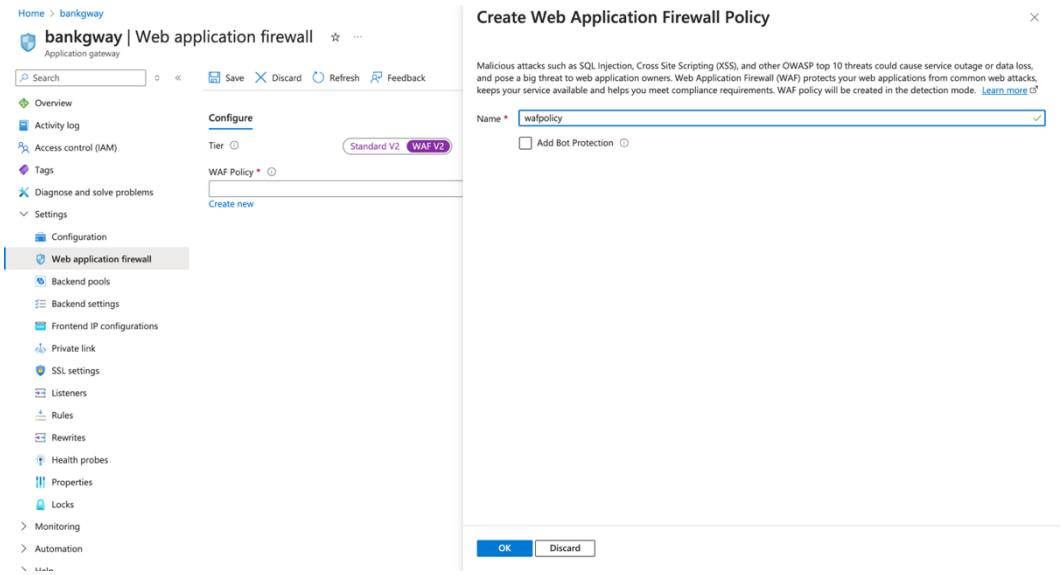


Slika 31 Azure Web App postavljanje mreže

5.2.2 Implementacija Web Application Firewall-a

Nakon kreiranja Application Gateway-a, u postavkama Web Application Firewall-a unutar gateway-a, odabran je WAF v2 tier, koji omogućava napredne sigurnosne funkcije i dodatne mogućnosti konfiguracije. Kreiran je novi WAF policy imena "webpolicy", koji je dizajniran za zaštitu web aplikacije od raznih sigurnosnih prijetnji kao što su SQL injection, cross-site scripting (XSS), i druge uobičajene web aplikacijske ranjivosti.

Nakon kreiranja WAF policy-a, dodijelili smo gaApplication Gateway-u kako bi se primijenio na sav promet koji prolazi kroz gateway. Ovaj korak je ključan jer osigurava da svi dolazni i odlazni zahtjevi budu provjereni i filtrirani prema sigurnosnim pravilima koja su zadano definirana u "webpolicy". Time je postignuto da web aplikacija ima robustnu zaštitu, čime se smanjuje rizik od sigurnosnih incidenata i povećava pouzdanost i sigurnost infrastrukture.



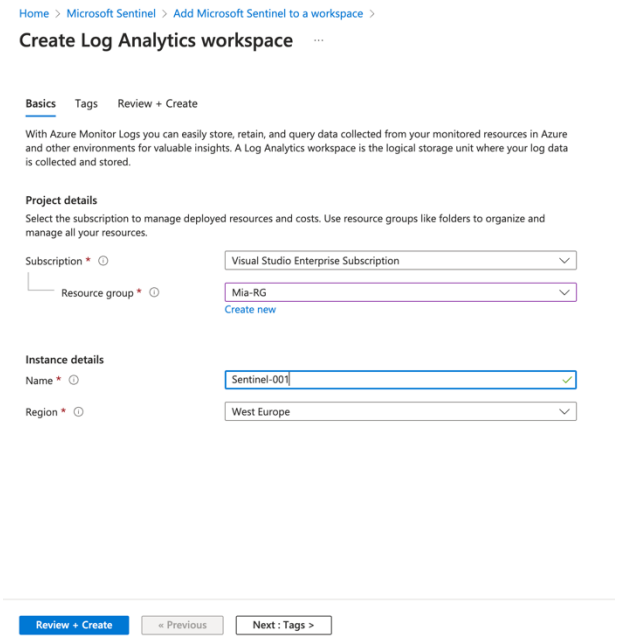
Slika 32 Implementacija Web Application Firewall-a

5.3 Implementacija Microsoft Sentinel-a

Praćenje i upravljanje sigurnosnim događajima (Security Information and Event Management - SIEM) ključni su aspekti sigurnosnih operacija. Ovi procesi omogućavaju organizacijama prepoznavanje, analizu i odgovor na sigurnosne incidente u stvarnom vremenu. U ovom odeljku, orijentirat ćemo se na moćan alat dostupan na Azure platformi - Microsoft Sentinel. Microsoft Sentinel je skalabilno, nativno rješenje u oblaku za SIEM (Security Information and Event Management) i SOAR (Security Orchestration Automated Response). Dizajniran je da omogući brzo otkrivanje i reagiranje na sigurnosne prijetnje.

5.3.1 Kreiranje Log Analytics radnog prostora

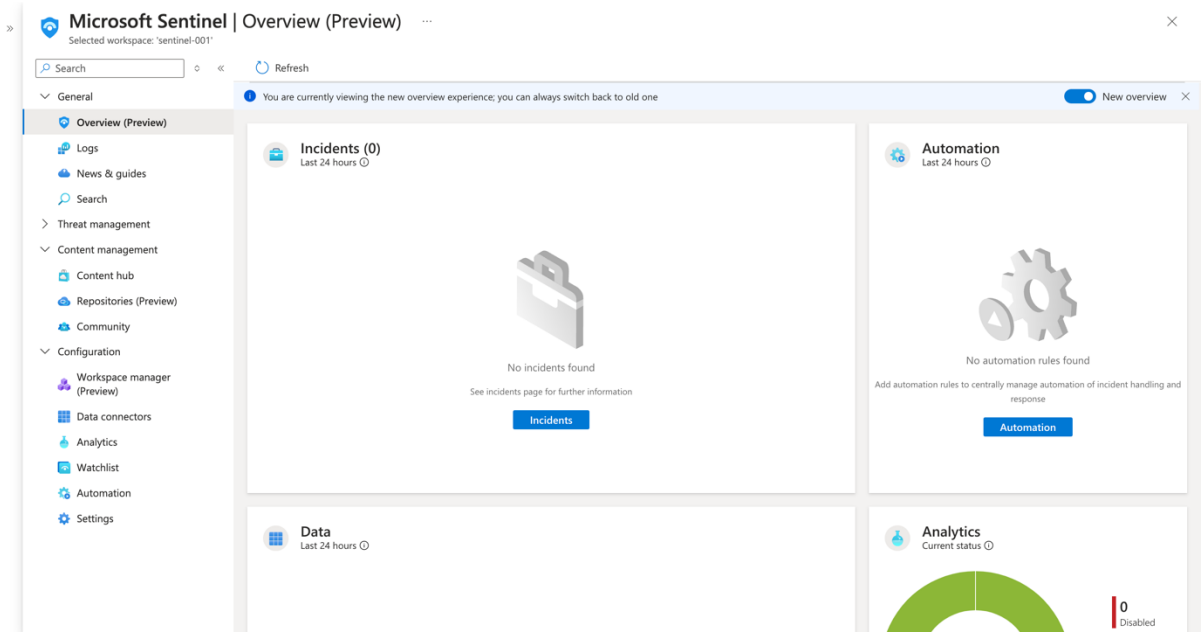
Microsoft Sentinel je instaliran u Log Analytics radni prostor. Većina aspekata implementacije fokusirana je na kreiranje Log Analytics radnog prostora, tako da implementacija započinje s kreiranjem Log Analytics radnog prostora. Dobra praksa prilikom implementacije Sentinel-a je kreirati vlastiti radni prostor, a ne koristiti već postojeći.



Slika 33 Kreiranje Log Analytics workspace-a

Nakon što je Log Analytics radni prostor kreiran, dodajemo ga u Microsoft Sentinel radni prostor. Ova konfiguracija nam omogućava da pratimo i upravljamo sigurnosnim događajima koristeći Microsoft Sentinel, koji koristi mogućnosti Log Analytics-a za prikupljanje i analizu podataka.

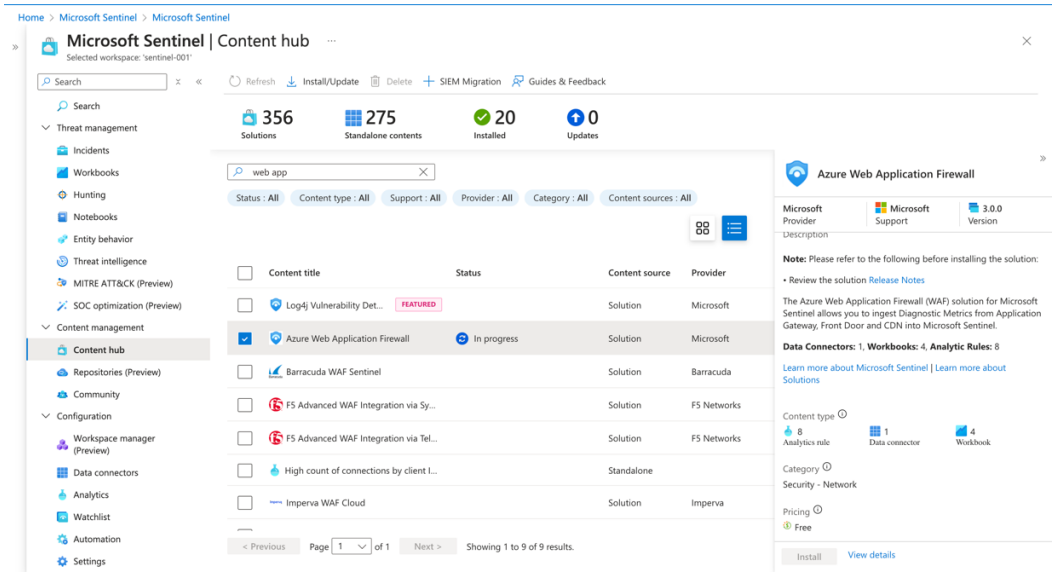
Microsoft Sentinel sam po sebi ne pruža nikakve podatke. Da bi bio funkcionalan, potrebno je konfigurirati konektore za podatke. Dostupne su stotine konektora, a odgovornost administratora je odabrati one najkorisnije. Svaki unos podataka ima svoju cijenu, pa je potrebno prioritizirati radi optimalne troškovne učinkovitosti. Kada ne postoje veze s drugim izvorima podataka, korisnik zapravo ne može ništa raditi sa Sentinel-om.



Slika 34 Microsoft Sentinel

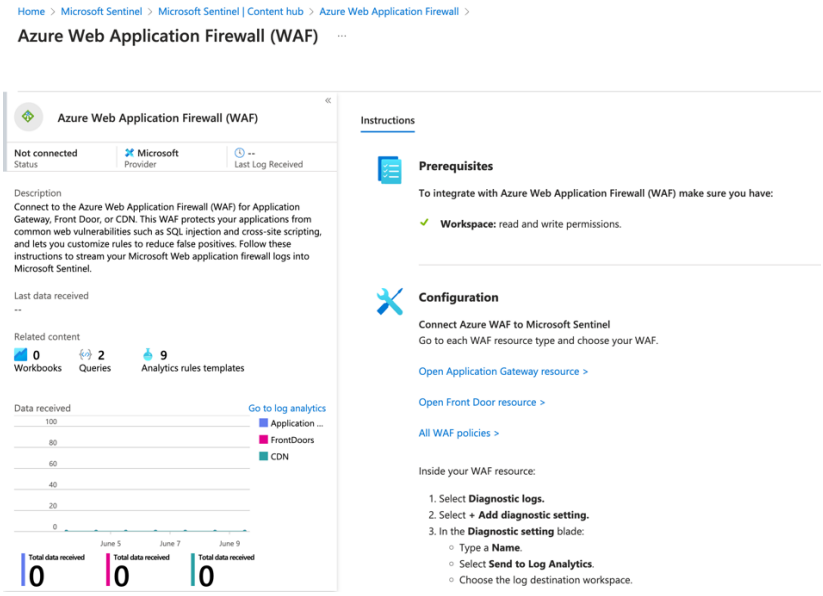
5.3.2 Podatkovni konektori

Nakon dodavanja radnog prostora u Microsoft Sentinel, sljedeći korak je korištenje konektora za podatke kako bi se počeli unositi podaci u Microsoft Sentinel. Konektori za podatke dostupni su kao dio rješenja iz sadržajnog čvorišta (Content Hub) u Microsoft Sentinel-u. Nakon što se rješenje instalira iz sadržajnog čvorišta, povezani konektori za podatke bit će dostupni za omogućavanje i konfiguraciju.



Slika 35 Sadržajni centar Microsoft Sentinel-a

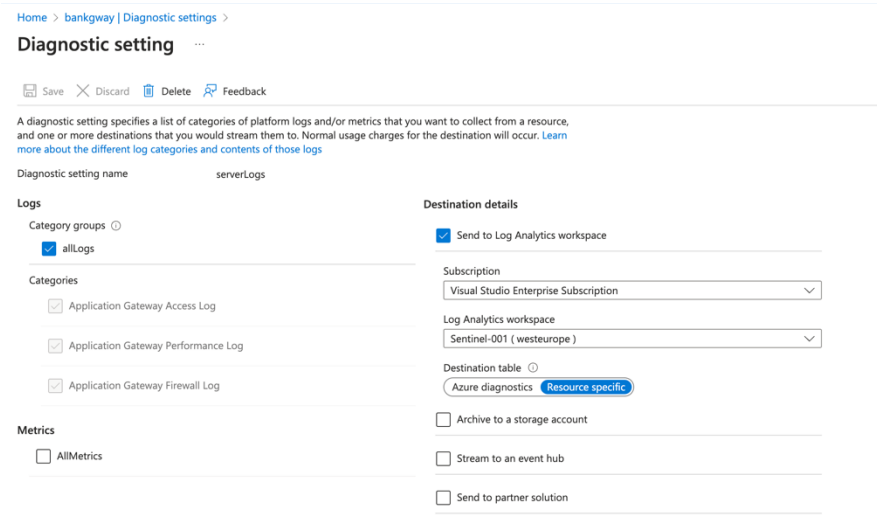
Svaki data konektor ima stranicu konektora, a kada se otvori, korisnik je dočekan s prikazom kao na slici 36. Konektori imaju određene preduvjete i upute za konfiguraciju. Preduvjeti obično zahtijevaju da korisnik ima potpuna administratorska prava za određenu data konekciju. U sljedećem koraku demonstrirana je implementacija na primjeru konektora za Azure Web Application Firewall.



Slika 36 Primjer WAF connector-a

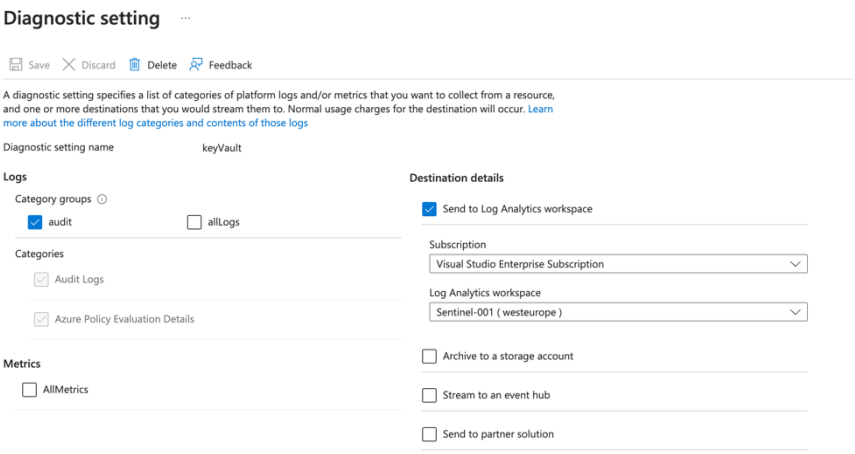
Azure Web Application Firewall (WAF) konektor omogućava nadzor i zaštitu web aplikacija od različitih vrsta cyber napada. Ovaj konektor prikuplja sigurnosne zapise i događaje s WAF-a te ih unosi u Microsoft Sentinel radi analize i korelacije s drugim sigurnosnim podacima. Pomoću ovog konektora možemo: analizirati promet koji prolazi kroz web aplikaciju, identificirati različite napade kao što su XSS, SQL injekcije i druge prijetnje te dobiti uvid u potencijane sigurnosne probleme i slabosti aplikacije

Kako bi povezali navedeni konektor sa resursom, u Application Gateway-u preko Diagnostic Settings-a šaljemo sljedeće kategorije logova: Application Gateway Access Log, koji bilježi podatke o svim zahtjevima koji prolaze kroz Application Gateway; Application Gateway Performance Log, koji pruža informacije o performansama Application Gateway-a; i Application Gateway Firewall Log, koji bilježi sve događaje povezane s Web Application Firewall (WAF) pravilima.



Slika 37 Slanje logova iz App Gateway-a u Log Analytics okruženje

Također je implementiran Azure Key Vault konektor koji omogućava sigurno pohranjivanje i upravljanje pristupom osjetljivim podacima kao što je implementirani connection string. Integracija s Azure Key Vault-om omogućava aplikaciji siguran pristup osjetljivim informacijama, a njihovo praćenje putem konektora omogućuje detaljan nadzor i audit pristupa tim podacima.



Slika 38 Slanje Key Vault logova u Analytics okruženje

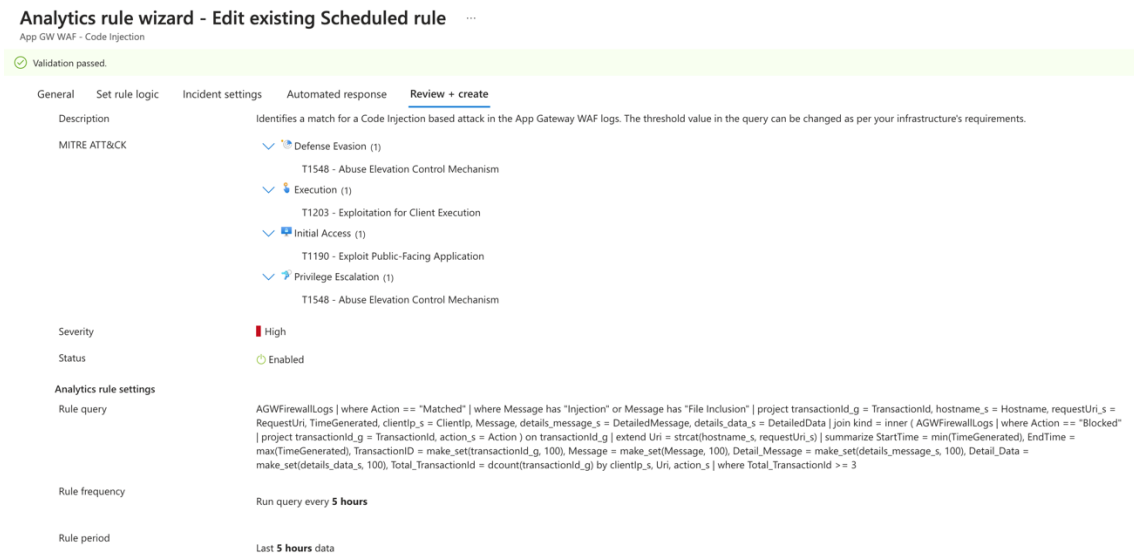
Ova dva konektora zajedno osiguravaju sveobuhvatno prikupljanje, analizu i upravljanje podacima i sigurnosnim događajima aplikacije, čineći ih idealnim odabirom za potrebe sigurnosnog i operativnog nadzora.

5.3.3 Analitička pravila

Da bismo počeli otkrivati prijetnje i neuobičajeno ponašanje, kreiramo prilagođena analitička pravila u izborniku analitike pod konfiguracijom. Analitička pravila služe za pretraživanje određenih događaja ili više povezanih skupova događaja u okruženju. Kada su uvjeti navedeni u pravilima ispunjeni, stvara se upozorenje ili incident. Sentinel nudi gotove predloške pravila koji samostalno funkcioniraju. Te gotove predloške pravila moguće je modificirati kako bismo kreirali prilagođena pravila prema specifičnim potrebama. Pravila se sastoje od logike pravila, postavki incidenta i automatskog odgovora. Upit se izvršava u određenom vremenskom razdoblju, na primjer, svakih sat vremena ili jednom dnevno.

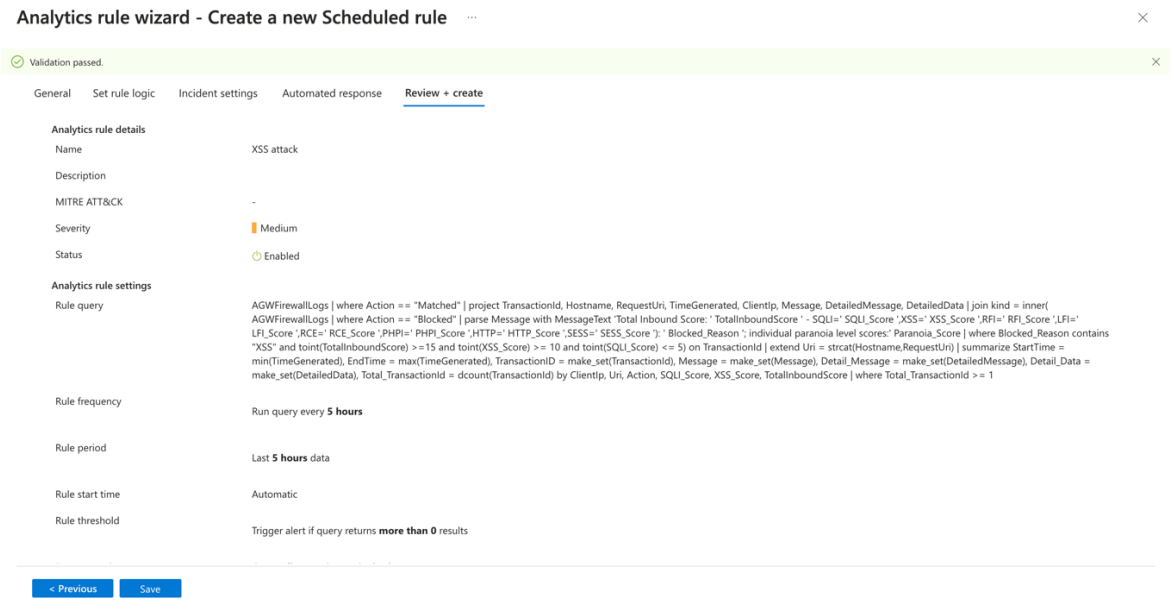
U ovom poglavlju implementirana su analitička pravila za praćenje dva najčešća napada - XSS napada i pristupa Azure Key Vaultu. Ova pravila omogućuju detekciju sumnjivih aktivnosti i potencijalnih prijetnji kako bi se osigurala sigurnost naše aplikacije. Implementacija aplikacije može nadograditi dodavanjem novih analitičkih pravila ili proširenjem postojećih kako bi se adresirale nove sigurnosne prijetnje ili scenariji. Ova prilagodljivost omogućuje kontinuirano poboljšanje sigurnosne strategije i zaštitu naše aplikacije od evoluirajućih prijetnji

Implementirano analitičko pravilo "App GW WAF - Code Injection" (Slika 39), osmišljeno je za otkrivanje potencijalnih napada temeljenih na ubacivanju koda u logovima Web Application Firewalla (WAF) na Application Gatewayu (AGW). Primjenom KQL upita, pravilo pretražuje logove i traži podudarajuće uzorke koji sugeriraju pokušaje ubacivanja koda ili uključivanje datoteka. Na primjer, provjerava se jesu li određene akcije logirane kao "Matched" te sadrže riječi poput "Injection" ili "File Inclusion". U slučaju pronalaska takvih uzoraka, pravilo analizira transakcije i identificira IP adrese, URI-je i akcije koje su bile uključene u te napade. Ako se identificira tri ili više transakcija s istim IP-om, URI-jem i akcijom, generira se upozorenje. Ovo pravilo postavlja visoki stupanj sigurnosti s obzirom na težinu napada koje može otkriti. Pokreće se svakih 5 sati i analizira podatke iz posljednjih 5 sati. Generira incidente na temelju upozorenja, omogućujući brzu reakciju na potencijalne sigurnosne prijetnje. Automatizirani odgovor na generirane incidente trenutno nije konfiguriran, ali pravilo omogućuje jednostavno proširenje s dodatnim koracima automatske obrane u budućnosti.



Slika 39 Implementirano analitičko pravilo za napad injekcije koda

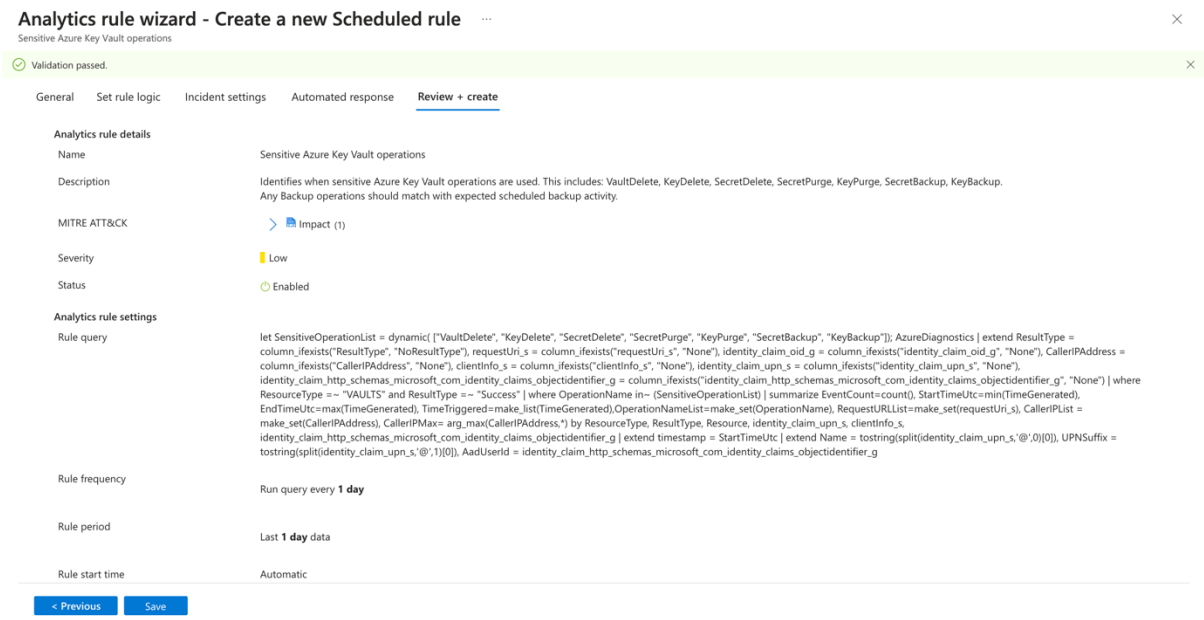
Analitičko pravilo "XSS attack" osmišljeno je za identifikaciju napada XSS (Cross-Site Scripting) putem Application Gateway Firewalla. Sam postupak detekcije provodi se analizom logova pristupa AGWFirewallLogs. Pravilo pretražuje zapise gdje je akcija označena kao "Matched", što ukazuje na situacije kada su potencijalni XSS napadi prepoznati. Zatim, vrši se spajanje tih zapisa s logovima gdje je akcija "Blocked", što označava da je napad blokiran. Dodatno, korištenjem parsiranja, pravilo provjerava detalje blokiranja kako bi identificiralo specifične karakteristike XSS napada, kao što su ocjene sigurnosnih rizika (Score) i razlozi blokiranja. Otkriveni napadi grupiraju se prema IP adresi klijenta, URI-ju (Uniform Resource Identifier) zahtjeva, akciji (Matched ili Blocked), te ocjenama sigurnosnih rizika kao što su ocjene SQL Injectiona, XSS-a te ukupna ocjena dolaznog prometa. Prag pravila postavljen je tako da se aktivira ako pronađe barem jedan napad XSS-a.



Slika 40 Implementirano analitičko pravilo za XSS napad

"Sensitive Azure Key Vault operations" ima za cilj identificirati i nadzirati osjetljive operacije koje se izvršavaju nad Azure Key Vaultom. Upit započinje definiranjem liste osjetljivih operacija koje uključuju brisanje, trajno brisanje i sigurnosno kopiranje ključeva i tajni. Nakon toga, filtrira dijagnostičke logove povezane s Azure Key Vault operacijama te proširuje informacije u logovima koristeći dodatna polja kao što su rezultat operacije, URI zahtjeva, IP adresa pozivatelja i identitet korisnika. Zatim se vrši filtriranje logova kako bi se identificirale samo operacije koje su dio liste osjetljivih operacija i koje su završile uspješno. Nakon

filtriranja, rezultati se sumiraju kako bi se prikazale ključne informacije o broju događaja, vremenu izvršenja, operacijama, URI-ju zahtjeva, IP adresama pozivatelja i identitetima korisnika. Dodatno, pravilo ekstrahira imena korisnika i njihove identifikatore za daljnju analizu. Konačno, pravilo postavlja prag za generiranje upozorenja ako se otkrije bilo koja od osjetljivih operacija i omogućuje stvaranje incidenata kako bi se dodatno istražile aktivnosti koje bi mogle predstavljati rizik za sigurnost organizacije.

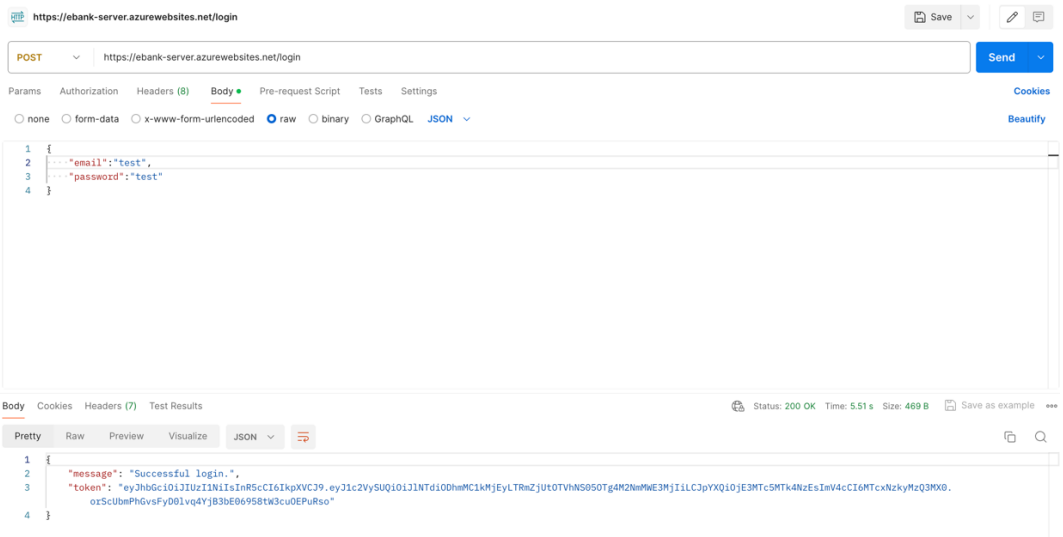


Slika 41 Implementirano analitičko pravilo za osjetljive radnje u Azure Key Vault-u

6 Testiranje sigurnosnih implementiranih mjera

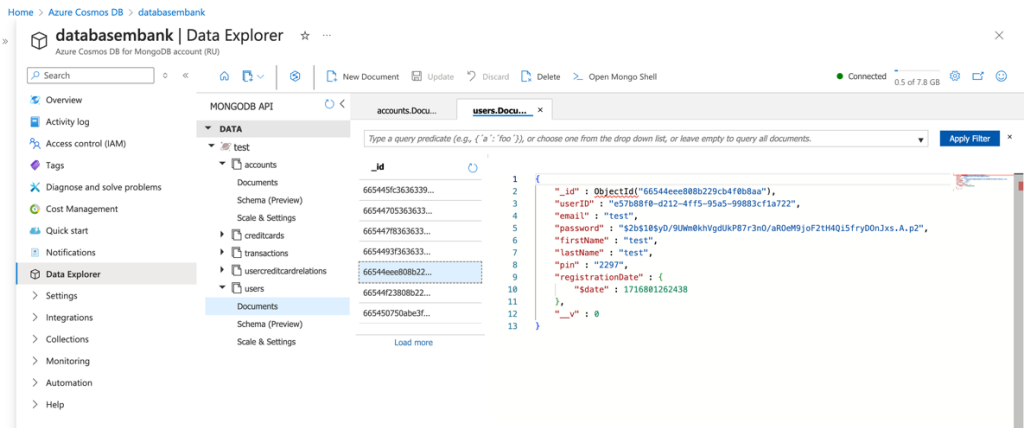
6.1 Testiranje Azure Key Vault-a

Testiranje dostupnosti Azure Key Vault-a je vrlo jednostavno. Koristeći Postman alat, šaljemo zahtjev za prijavu korisnika na server putem portala. Ako dobijemo uspješnu prijavu, to znači da smo uspješno konfigurirali našu tajnu, odnosno connection string baze podataka u Azure Key Vault. Time je connection string sigurno spremljen u Azure Key Vault.



Slika 42 Uspješno prijavljeni korisnik – testiranje Key Vault-a

Dodatno, u Data Exploreru baze podataka može se vidjeti spremljenog korisnika koji je uspješno prijavljen.



Slika 43 Korisnik spremljen u bazu podataka - testiranje Key Vault-a

6.2 Testiranje Azure Application Gateway-a i WAF-a kroz analizu logova

U ovom pristupu, koristit ćemo logove pristupa (Access Logs) kako bismo analizirali ispravnost rada Azure Application Gateway-a i Web Application Firewall-a (WAF). Umjesto izvođenja aktivnih testova ili simulacija napada, oslonit ćemo se na stvarne podatke o prometu kako bismo identificirali eventualne probleme ili sigurnosne prijetnje.

U analizi logova pristupa, fokusiramo se na prikupljanje, filtriranje i interpretaciju podataka kako bismo ocijenili ispravnost funkcioniranja Azure Application Gateway-a i Web Application Firewall-a (WAF-a). Prvenstveno prikupljamo relevantne logove iz ovih servisa za pohranu. Zatim filtriramo ove podatke kako bismo izdvojili ključne informacije poput IP adresa, URL-ova, korisničkih agenata i HTTP status kodova. Kroz analizu tih filtriranih podataka, tražimo uzorke u prometu koji bi mogli ukazivati na probleme ili sigurnosne prijetnje. Na primjer, identificiramo ponovljene neuspješne pokušaje pristupa, koji mogu sugerirati pokušaje brute force napada. Kategoriziramo probleme prema ozbiljnosti, pružajući posebnu pozornost na HTTP status kodove koji ukazuju na moguće sigurnosne prijetnje ili konfiguracijske probleme. Dodatno, provjeravamo odgovore firewalla na određene IP adrese kako bismo osigurali da je konfiguracija efikasna u detekciji i blokiranju malicioznih aktivnosti, potvrđujući tako ispravnost i učinkovitost WAF-a.

6.2.1 Prikupljanje i pristup logovima

Prvi korak u analizi logova je prikupljanje i pristup logovima generiranim od strane Azure Application Gateway-a i WAF-a. Kojima možemo pristupiti iz samog resursa ili putem Log Analytics radnog okruženja.

```
1  AGWAccessLogs
```

Results | Chart

TimeGenerated [UTC]	OperationName	ListenerName	RuleName	BackendPoolName	BackendSettingName	InstanceId	ClientIp	HttpMethod
5/23/2024, 6:28:32.000 AM	ApplicationGatewayAccess	httpsListener	myRoutingRule	myBackendPool	myBackendSetting	appgw_2	65.49.1.33	GET
5/23/2024, 6:22:21.000 AM	ApplicationGatewayAccess	httpsListener	myRoutingRule	myBackendPool	myBackendSetting	appgw_2	65.49.1.30	GET
5/23/2024, 6:22:05.000 AM	ApplicationGatewayAccess	httpsListener	myRoutingRule	myBackendPool	myBackendSetting	appgw_3	65.49.1.36	GET
5/23/2024, 6:20:45.000 AM	ApplicationGatewayAccess	httpsListener	myRoutingRule	myBackendPool	myBackendSetting	appgw_2	65.49.1.25	GET
5/23/2024, 6:16:25.000 AM	ApplicationGatewayAccess	httpsListener	myRoutingRule	myBackendPool	myBackendSetting	appgw_3	65.49.1.26	GET
5/23/2024, 6:15:55.000 AM	ApplicationGatewayAccess	httpsListener	myRoutingRule	myBackendPool	myBackendSetting	appgw_2	65.49.1.29	HEAD
5/23/2024, 6:13:59.000 AM	ApplicationGatewayAccess	httpsListener	myRoutingRule	myBackendPool	myBackendSetting	appgw_2	65.49.1.26	GET
5/23/2024, 5:20:47.000 AM	ApplicationGatewayAccess					appgw_3	127.0.0.1	
5/23/2024, 5:20:47.000 AM	ApplicationGatewayAccess					appgw_3	127.0.0.1	
5/23/2024, 5:18:51.000 AM	ApplicationGatewayAccess					appgw_2	127.0.0.1	
5/23/2024, 5:18:51.000 AM	ApplicationGatewayAccess					appgw_2	127.0.0.1	
5/23/2024, 5:18:31.000 AM	ApplicationGatewayAccess	httpsListener	myRoutingRule	myBackendPool	myBackendSetting	appgw_3	35.195.123.144	GET
5/23/2024, 3:16:01.000 AM	ApplicationGatewayAccess	httpsListener	myRoutingRule	myBackendPool	myBackendSetting	appgw_3	209.141.42.151	GET
5/23/2024, 2:40:37.000 AM	ApplicationGatewayAccess	httpsListener	myRoutingRule	myBackendPool	myBackendSetting	appgw_2	111.7.96.174	GET
5/23/2024, 2:39:26.000 AM	ApplicationGatewayAccess	httpsListener	myRoutingRule	myBackendPool	myBackendSetting	appgw_2	123.160.221.1...	GET

Slika 44 Prikupljeni logovi

6.2.2 Filtriranje logova prema neuspješnoj prijavi

Kako bismo identificirali potencijalne sigurnosne prijetnje, filtrirat ćemo logove kako bismo izdvojili neuspješne pokušaje prijave.

```
1  AGWAccessLogs
2  | where RequestUri =="/login"
3  | summarize FailedAttempts = count() by ClientIp
```

Results | Chart

ClientIp	FailedAttempts
54.198.55.229	5
54.156.251.192	2
34.235.48.77	2
93.143.31.226	2
95.168.124.21	162
93.143.114.131	2
84.54.51.13	1
93.143.50.0	11

Slika 45 Filtriranje logova prema neuspješnoj prijavi

IP adresa 95.168.124.21 ima znatno veći broj neuspjelih pokušaja prijave (162) u usporedbi s ostalim IP adresama. Ovo ukazuje na mogućnost pokušaja brutalne sile, gdje se sustavno isprobavaju različite lozinke u nadi da će se pogoditi ispravna kombinacija. Također, može se raditi o automatiziranim pokušajima prijave, gdje računalni program pokušava pristupiti sustavu koristeći unaprijed definirani skup vjerodajnica.

Nekoliko drugih IP adresa također pokazuje višestruke neuspjele pokušaje prijave, iako ne u tolikoj mjeri kao IP adresa 95.168.124.21. Na primjer, IP adresa 54.198.55.229 zabilježila je 5 neuspjelih pokušaja, dok su IP adrese 54.156.251.192, 34.235.48.77, 93.143.31.226, i 93.143.114.131 zabilježile po 2 neuspjela pokušaja svaka. IP adresa 93.143.50.0 imala je 11 neuspjelih pokušaja, a IP adresa 84.54.51.113 imala je 1 neuspjeli pokušaj. Iako je broj ovih pokušaja manji u odnosu na IP adresu s najviše pokušaja, ponovljeni neuspjesi mogu ukazivati na pokušaje pristupa sustavu koji su u tijeku ili koji su ponovljeni više puta.

6.2.3 Analiza sumnjivih adresa

U odnosu na tablicu, uzet ćemo IP adresu s najvećim brojem pokušaja prijave i pogledati u `AGWFirewallLogs` koji generira implementirani firewall u gatewayu, kako bismo istražili što osoba s tom IP adresom pokušava i kako je firewall reagirao na te pokušaje.

```
1  AGWFirewallLogs
2  | where ClientIp=="95.168.124.21"
3  | project TimeGenerated, ClientIp, RequestUri,  Action, Message
4
```

Results | Chart

TimeGenerated [UTC]	ClientIp	RequestUri	Action	Message
5/18/2024, 9:37:35.000 PM	95.168.124.21	/login	Blocked	Inbound Anomaly Score Exceeded (Total Score: 9)
5/18/2024, 9:37:35.000 PM	95.168.124.21	/login	Matched	Request Containing Content Requires Content-Type header
5/18/2024, 9:37:35.000 PM	95.168.124.21	/login	Matched	Missing User Agent Header
5/18/2024, 9:37:35.000 PM	95.168.124.21	/login	Matched	Request Containing Content, but Missing Content-Type header
5/18/2024, 9:37:35.000 PM	95.168.124.21	/login	Blocked	Inbound Anomaly Score Exceeded (Total Score: 9)
5/18/2024, 9:37:35.000 PM	95.168.124.21	/login	Matched	Request Containing Content Requires Content-Type header
5/18/2024, 9:37:35.000 PM	95.168.124.21	/login	Matched	Missing User Agent Header
5/18/2024, 9:37:35.000 PM	95.168.124.21	/login	Matched	Missing User Agent Header
5/18/2024, 9:37:35.000 PM	95.168.124.21	/login	Matched	Request Containing Content, but Missing Content-Type header
5/18/2024, 9:37:35.000 PM	95.168.124.21	/login	Blocked	Inbound Anomaly Score Exceeded (Total Score: 9)
5/18/2024, 9:37:35.000 PM	95.168.124.21	/login	Matched	Request Containing Content Requires Content-Type header
5/18/2024, 9:37:35.000 PM	95.168.124.21	/login	Matched	Request Containing Content, but Missing Content-Type header
5/18/2024, 9:37:36.000 PM	95.168.124.21	/login	Blocked	Inbound Anomaly Score Exceeded (Total Score: 9)
5/18/2024, 9:37:36.000 PM	95.168.124.21	/login	Matched	Request Containing Content Requires Content-Type header
5/18/2024, 9:37:36.000 PM	95.168.124.21	/login	Matched	Missing User Agent Header

Slika 46 Filtriranje firewall logova prema sumnjivoj IP adresi

Primjeri logova iz AGWFirewallLogs za IP adresu 95.168.124.21

Log 1:

- Vrijeme generiranja [UTC]: 2024-05-18T21:37:35Z
- IP adresa klijenta: 95.168.124.21
- Zahtjevni URI: /login
- Akcija: Blocked
- Poruka: Inbound Anomaly Score Exceeded (Total Score: 9)

U ovom zapisu, firewall je blokirao pokušaj pristupa zbog prekoračenja dozvoljenog anomalijskog skora. Ukupni skor anomalija iznosi 9, što je dovoljno visoko da pokrene blokiranje.

Log 2:

- Vrijeme generiranja [UTC]: 2024-05-18T21:37:35Z
- IP adresa klijenta: 95.168.124.21
- Zahtjevni URI: /login
- Akcija: Matched
- Poruka: Request Containing Content Requires Content-Type header

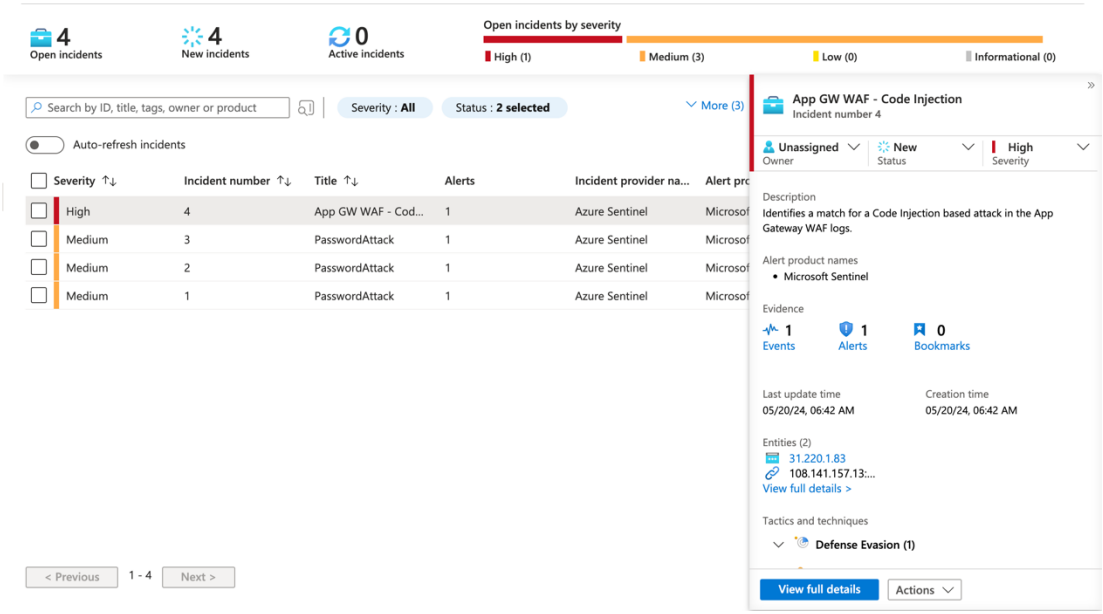
U ovom zapisu, firewall je prepoznao zahtjev koji sadrži sadržaj, ali nedostaje Content-Type header. Iako ovaj pokušaj nije blokiran, označen je kao sumnjiv.

IP adresa 95.168.124.21 pokazuje značajnu aktivnost s višestrukim neuspjelim pokušajima prijave. Ovo može ukazivati na potencijalni brute force napad ili druge zlonamjerne aktivnosti. Firewall je uspješno detektirao i blokirao sumnjive aktivnosti na temelju visokog anomalijskog skora, kao i bilježenje zahtjeva koji nisu potpuno ispunjavali sigurnosne standarde (npr. nedostatak Content-Type ili User Agent headera).

6.3 Testiranje Microsoft Sentinela

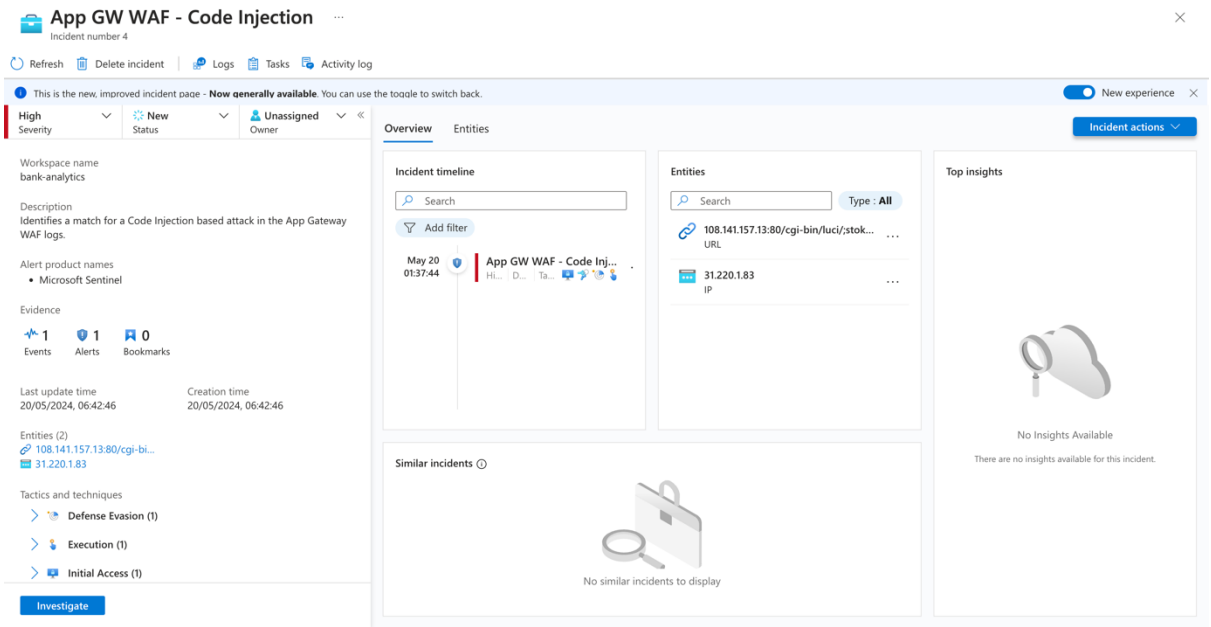
Implementirali smo analitičko pravilo za napade ubrizgavanja koda (Code Injection) u Microsoft Sentinel kako bismo testirali funkcionalnost i učinkovitost sustava (5. poglavlje). Nakon implementacije, Microsoft Sentinel je uspješno detektirao incident povezan s ovim napadom, što je vidljivo iz prikazanih slika.

Na prvoj slici možemo vidjeti sučelje Microsoft Sentinela koje prikazuje trenutno stanje sigurnosnih incidenata, uključujući broj otvorenih incidenata (4), novih incidenata (4) i aktivnih incidenata (0). Incidenti su klasificirani prema ozbiljnosti (High, Medium, Low), a konkretno je prikazan incident visokog prioriteta s naslovom "App GW WAF - Code Injection".

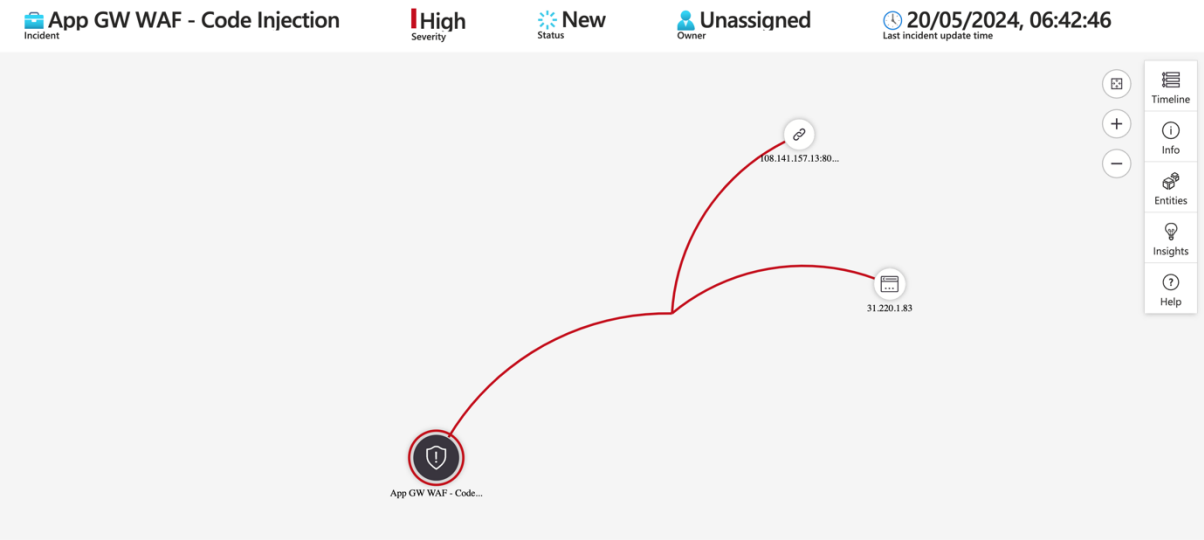


Slika 47 Microsoft Sentinel pokrenuti incidenti

Opis incidenta identificira napad temeljen na ubrizgavanju koda u App Gateway WAF logovima. Prikazani su entiteti povezani s incidentom, uključujući specifične IP adrese i URL-ove. Vremenska linija incidenta prikazuje redoslijed događaja vezanih uz incident, dok su taktičke informacije kategorizirane kao Defense Evasion, Execution i Initial Access.



Slika 48 Microsoft Sentinel - aktiviran Code Injection



Slika 49 Microsoft Sentinel - vizualizacija incidenta

Treća slika prikazuje grafički prikaz incidenta, omogućujući vizualizaciju povezanosti različitih entiteta (IP adresa i URL) s incidentom. Ova vizualizacija pomaže u razumijevanju odnosa između različitih elemenata incidenta.

Ovo testiranje potvrđuje da implementacija Microsoft Sentinela radi ispravno te učinkovito detektira i prijavljuje sigurnosne prijetnje. Kroz detalje o incidentu, uključujući povezane entitete, taktičke informacije i kronologiju događaja, te kroz vizualizaciju povezanosti, jasno je da Microsoft Sentinel omogućuje timovima za sigurnost da brzo i precizno reagiraju na incidente. Ovom implementacijom se mogu realizirati daljnji koraci analize prikupljenih podataka i poduzimanje odgovarajućih mjera za otklanjanje prijetnje i poboljšanje sigurnosnih postavki.

7 Zaključak

Računarstvo u oblaku, posebno kroz Microsoft Azure, nudi izuzetne mogućnosti za skalabilnost, fleksibilnost i efikasnost u radu s podacima i aplikacijama. Međutim, sigurnost ostaje ključni izazov. Korištenje Microsoft Azure platforme za računarstvo u oblaku transformira način na koji organizacije upravljaju podacima i aplikacijama, omogućavajući im da iskoriste nevjerojatne prednosti u skalabilnosti, fleksibilnosti i efikasnosti. Međutim, kako bi ove prednosti bile potpuno ostvarene, sigurnost mora biti glavni prioritet. Uvođenje sigurnosnih mjera kao što su Azure Key Vault, Azure Application Gateway sa Web Application Firewall-om (WAF) i Microsoft Sentinel predstavlja ključne korake u osiguravanju zaštite podataka i aplikacija. Redovito ažuriranje sigurnosnih mjera i kontinuirano praćenje potencijalnih prijetnji ključni su za održavanje sigurnosti u oblačnom okruženju. Automatizacija sigurnosnih procesa može značajno ubrzati otkrivanje i odgovaranje na incidente, smanjujući vrijeme potrebno za reakciju i minimizirajući potencijalnu štetu. Edukacija zaposlenih o najboljim praksama za sigurnost informacija također igra vitalnu ulogu u jačanju ukupne sigurnosne strategije, jer ljudski faktor može biti najslabija karika u sigurnosnom lancu. Proaktivna analiza i revizija sigurnosnih postavki pomažu u identifikaciji i otklanjanju potencijalnih ranjivosti prije nego što one budu iskorištene. Integracija dodatnih sigurnosnih alata i usluga može pružiti dodatne slojeve zaštite, osiguravajući sveobuhvatnu sigurnost IT infrastrukture u oblaku. Korištenjem naprednih alata za otkrivanje i prevenciju upada, enkripcijom podataka i drugim sigurnosnim rješenjima, organizacije mogu dodatno učvrstiti svoju obranu protiv sofisticiranih prijetnji. Iako su tehničke mjere od izuzetne važnosti, ključ uspjeha leži i u kulturi sigurnosti unutar organizacije. Potrebno je osigurati da svi zaposlenici razumiju važnost sigurnosti i svoje uloge u održavanju sigurnog okruženja. Redovite obuke i osvježavanje znanja o sigurnosnim praksama su neophodni kako bi se osiguralo da sigurnost ostane na visokom nivou. Microsoft Azure, uz pravilno planiranje i implementaciju naprednih sigurnosnih mjera, može pružiti sigurno i efikasno rješenje za moderne aplikacije i podatke. Organizacije koje posvete pažnju kontinuiranom unapređenju sigurnosti mogu značajno smanjiti rizik od sigurnosnih incidenata, osigurati kontinuitet poslovanja i maksimalno iskoristiti sve prednosti koje računarstvo u oblaku nudi.

Literatura

[1] Mell, P., & Grance, T. (2011). The NIST definition of cloud computing. National Institute of Standards and Technology

[2] Barr, J., Marathe, A., & Peterson, L. (2016). Cloud Computing: Concepts, Technology & Architecture. Morgan Kaufmann

[3] Borra, P. (2024). Comparative Review: Top Cloud Service Providers ETL tools - AWS vs. Azure vs. GCP

[4] Mather, T., Kumaraswamy, S., & Latif, S. (2009). Cloud Security and Privacy: An Enterprise Perspective on Risks and Compliance

[5] Microsoft Azure Documentation https://docs.microsoft.com/en-us/azure/

[6] GeeksforGeeks MERN Stack vs Other Stacks: A Comparative Analysis https://www.geeksforgeeks.org/mern-stack-vs-other-stacks-a-comparative-analysis/

[7] Remotely.works Blog .NET vs MEAN vs MERN: Which Development Stack is Right for You? https://www.remotely.works/blog/net-vs-mean-mern-which-development-stack-is-right-for-you

[8] Grobauer, B., Walloschek, T., & Stocker, E. (2011). Understanding cloud computing vulnerabilities

[9] Rittinghouse, J. W., & Ransome, J. F. (2017). Cloud Computing: Implementation, Management, and Security

Tablica slika