Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.738.5:339
DOI: https://doi.org/10.24867/03BE27Djanic

# PROVERA IDENTITETA I AUTORIZACIJA KORISNIKA U MICROSOFT AZURE PLATFORMI

# AUTHENTICATION AND ACCESS CONTROL WITHIN MICROSOFT AZURE PLATFORM

Dražen Đanić, *Fakultet tehničkih nauka, Novi Sad*

**Oblast – ELEKTROTEHNIKA I RAČUNARSTVO**

**Kratak sadržaj –** *U radu je na primeru softverskog sistema za rezervaciju u restoranima opisano korišćenje platforme Microsoft Azure za potrebe rešavanja problema provere identiteta i autorizacije korisnika. Opis sistema obuhvata opis njegovog modela i implementacije.*

**Ključne reči:** *Provera identiteta, Autorizacija, Microsoft Azure, Azure Active Directory, Softverski sistem*

**Abstract** – *This document describes features of Microsoft Azure for authentication and authorization of users. These features are demonstrated on a system for managing reservations in restaurants. The system is described through its model and implementation.*

**Keywords:** *Authentication, Authorization, Microsoft Azure, Azure Active Directory, Software system*

## 1. UVOD

U svetu IT-a (Informacione tehnologije) je sve veći trend da se obezbeđivanje računarskih resursa, hardvera i softvera, ostvaruje putem takozvanog *cloud computing*-a (računarstvo u oblacima). *Cloud computing* se definiše kao model za isporuku računarskih resursa korisnicima u vidu servisa. Ta isporuka se obavlja preko odgovarajuće računarske mreže i to po prijemu zahteva. Sistem iz kojeg se pružaju traženi resursi nosi naziv *cloud*. Tako na primer, *cloud computing*-om se može dobiti pristup različitim aplikacijama, elementima za razvoj i isporuku aplikacija, skladištima podataka itd. Upotrebom *cloud computing*-a se potencijalno može povećati produktivnost, povećati pouzdanost, smanjiti finansijski troškovi itd. Među čitavim nizom *cloud* sistema koji postoje na tržištu i koji su dostupni širokim masama korisnika nalazi se i *Microsoft Azure*, *cloud* koji je u vlasništvu kompanije *Microsoft* [1, 2, 3].

Pojedine aplikacije mogu imati potrebu za izvršavanjem procesa provere identiteta (autentifikacije) i autorizacije. Autentifikacija predstavlja proces kojim se proverava identitet nekog entiteta, odnosno proverava se da li je entitet zaista onaj za koji tvrdi da jeste. Autorizacija predstavlja proces kojim se utvrđuje da li je nekom entitetu dozvoljen pritup traženom resursu. Autorizacija zahteva uspešno izvršenu autentifikaciju. Treba imati na umu da entitet može biti korisnik, aplikacija ili nešto slično.

---

**NAPOMENA:**
**Ovaj rad proistekao je iz master rada čiji mentor je bio dr Goran Sladić, vanr. prof.**

Implementacija pomenutih procesa u pojedinim situacijama može biti prilično složena i može oduzeti puno vremena i energije koje bi se inače mogle iskoristiti za implementaciju ključnih elemenata aplikacije, odnosno elemenata koji predstavljaju poslovnu logiku same aplikacije. Zbog toga se sve više javlja potreba da se obavljanje autentifikacije i autorizacije ostvari upotrebom nekog servisa koji je za to specijalizovan. Jedan takav servis jeste *Azure* AD (*Active Directory*), servis koji nudi *Microsoft Azure* [4, 5, 6].

U ovom radu je razmatrano obavljanje procesa autentifikacije i autorizacije korisnika upotrebom mogućnosti koje pruža *cloud Microsoft Azure*, odnosno upotrebom servisa *Azure* AD. Primena servisa *Azure* AD je pokazana na primeru jednog softverskog sistema. Ovaj softverski sistem, između ostalog, omogućava rezervaciju stolova u restoranima, a autentifikaciju i autorizaciju korisnika ostvaruje upotrebom *Azure* AD-a. Sistem u svoj sastav uključuje dve komponente, od kojih je jedna implementirana kao SPA (*Single Page Application*), a druga kao *web* API (*Application Programming Interface*).

## 2. MICROSOFT AZURE

*Microsoft Azure* je javni *cloud* [1] koji nudi niz servisa koje klijent može iskoristiti za razvijanje, testiranje, isporučivanje i upravljanje aplikacijama na kojima radi. Pomoću tehnologija koje nudi *Microsoft*, klijent može kreirati hibridni *cloud* [1] u čiji sastav bi ušao i *Microsoft Azure* [3].

Microsoft Azure klijentima pruža servise preko međusobno povezanih datacenter-a (centar podataka, centar koji se sastoji od velikog broja povezanih računarskih resursa) smeštenih u različitim regionima sveta. Od klijenta traži da u slučaju određenih servisa navede region koji odgovara datacenter-u iz kojeg želi da mu se pruži željeni servis. Nije sve jedno iz kojeg regiona se obezbeđuje servis jer udaljenost utiče na latenciju mrežnog zahteva. Zbog toga je veoma bitno da region iz kojeg se pruža neki servis krajnjem entitetu bude što bliži tom entitetu.

Klijentu su data na raspolaganje, korišćenje, različita sredstva pomoću kojih može da upravlja servisima koje *Microsoft Azure* pruža. Tako servisima može da upravlja upotrebom *web* aplikacije *Azure Portal*, kojoj se pristupa preko *web browser*-a (aplikacija za pregledanje *web* sadržaja). Pored upravljanja servisima, *Azure Portal* omogućava i uvid u informacije vezane za plaćanje pruženih usluga. Većinu poslova klijent može da obavi i programskim putem upotrebom odgovarajućeg REST

API-a. Inače, i sam *Azure Portal* koristi skup REST API-a. Za pojedine platforme su obezbeđeni SDK-ovi (*Software development kit*) koje klijent može da upotrebi da programskim putem upravlja servisima. Među platformama za koje su obezbeđeni SDK-ovi nalaze se: .NET, Node.js, Java, PHP, Python i Ruby. Za upravljanje servisima se mogu upotrebiti i alatke *Azure Command-Line Interface* i *Azure PowerShell*.

### 2.1. Azure AD

*Azure AD* je jedan od mnogobrojnih servisa koje *Microsoft Azure* nudi svojim klijentima. On se definiše kao servis koji omogućava nekoj organizaciji da upravlja identitetima i pristupanjima. Njegovom upotrebom organizacija može povećati produktivnost svojih korisnika i može povećati sigurnost svojih resursa.

Klijent upotrebom *Azure* AD-a može da za svakog korisnika obezbedi po jedan jedinstveni identitet, a zatim i da upravlja tim identitetima. Korisniku se može omogućiti da identitet koji mu je dodeljen iskoristi kako bi ostvario SSO pristup određenim aplikacijama. Te aplikacije mogu biti one aplikacije koje obezbeđuje sama organizacija kojoj pripada ili one aplikacije koje obezbeđuju neke druge organizacije kroz servise SaaS [1] tipa. *Azure* AD omogućava da se korisniku dozvoli da sam poništi zaboravljenu lozinku kako bi postavio novu i time izbegne potrebu da se obrati administratoru.

Zahvaljujući svemu tome, korisnik može da poveća kako svoju produktivnost tako i produktivnost same organizacije. Klijentu je data mogućnost da korisnike grupiše shodno potrebama kako bi njima lakše upravljao. Tako se, na primer, korisniku može automatski omogućiti ili zabraniti pristup nekoj aplikaciji na osnovu grupe kojoj pripada. *Azure* AD ima podršku za RBAC (*Role Based Access Control*) pa se nadležnosti i mogućnosti korisnika mogu odrediti dodeljivanjem odgovarajućih uloga i dozvola.

Sigurnost identiteta korisnika, aplikacija i ostalih resursa se može podići na viši nivo upotrebom multi-faktorske autentifikacije koja je podržana od strane *Azure* AD-a. Treba napomenuti da se sigurnost može povećati i upotrebom drugih mehanizama koje *Azure* AD nudi. Među njima se nalazi onaj koji daje mogućnost da se kontroliše kako neki korisnik može pristupiti nekoj aplikaciji povezanoj sa *Azure* AD za koju mu je dato pravo korišćenja. To kontrolisanje klijent može ostvariti definisanjem niza *conditional access policy*-a, odnosno definisanjem politike kojom se određuje načina na koji *Azure* AD treba da reaguje na pokušaj pristupa koji zadovoljava određene uslove.

Za *Azure* AD se vezuju dva pojma: *Azure* AD *tenant* i *Azure* AD *directory*. *Azure* AD *tenant* ili kraće *tenant* (zakupac, stanar) označava instancu Azure AD servisa koja je kreirana za potrebe jedne organizacije. Svaki *tenant* je izolovan i odvojen od ostalih *tenant*-a. To znači i da između *tenant*-a ne postoji odnos roditelj-dete, tj. ne postoji hijerarhija *tenant*-a. Svaki *tenant* ima svoj *Azure* AD *directory*, odnosno direktorijum u koji se smeštaju informacije o samom *tenant*-u i organizaciji. Tako se u *Azure* AD *directory*-u između ostalog mogu naći informacije o korisnicima, grupama i aplikacijama. Obično se u literaturi i praksi pojmovi *tenant*, *Azure* AD *directory* i *Azure* AD izjednačavaju, odnosno koriste se kao sinonimi.

### 2.2. Autentifikacija i autorizacija korišćenjem Azure AD-a

Klijenti servis Azure AD mogu iskoristiti da aplikacijama koje razvijaju obezbede mogućnost obavljanja autentifikacije i autorizacije. Zahvaljujući tome, klijenti ne moraju da implementiraju čitav mehanizam koji je neophodan za sigurno obavljanje takvih procesa. To, između ostalog, doprinosi značajnom pojednostavljenju implementacija aplikacija. Takođe, klijenti mogu veću pažnju posvetiti implementaciji poslovnih logika samih aplikacija.

Svaka aplikacija koja ostvaruje autentifikaciju i autorizaciju korišćenjem Azure AD-a mora biti registrovana u Azure AD dictionary-u odgovarajuće organizacije. Registrovanje je potrebno izvršiti kako bi Azure AD postao svestan te aplikacije i kako bi mu se obezbedile neophodne informacije za komunikaciju sa tom aplikacijom prilikom obavljanja autentifikacije i autorizacije ili prilikom razmene tokena. Takođe, registrovanje aplikacije se obavlja kako bi joj se obezbedile neophodne dozvole za pristupanje podacima u direktorijumu organizacije, drugim aplikacijama organizacije itd.

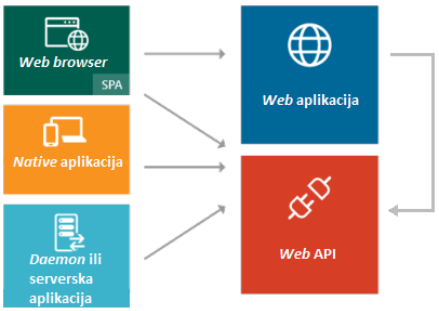
Aplikacije koje koriste *Azure* AD se mogu svrstati u jednu od dve kategorije: *single-tenant* aplikacije i *multi-tenant* aplikacije. *Single-tenant* aplikacije su one aplikacije koje su namenjene za korišćenje u samo jednoj organizaciji. Stoga one treba da budu registrovane u samo jednom *Azure* AD *dictionary*-u i to onom koji odgovara organizaciji kojoj je namenjen. *Multi-tenant* aplikacije su aplikacije koje su namenjene za korišćenje u više različitih organizacija, a ne samo u onoj u kojoj su inicijalno registrovane. Organizacija koja želi da koristi aplikaciju takve vrste mora dati saglasnost da bude registrovana i u njenom *Azure* AD *dictionary*-u.

Na slici 1 prikazani su različiti scenariji koje *Microsoft* navodi kao podržane scenarije od strane *Azure* AD-a, odnosno scenariji u kojima se *Azure* AD može koristiti za autentifikaciju i autorizaciju. Klijent nije ograničen na izbor programskih jezika i platformi koje će koristiti prilikom implementacije nekog od tih scenarija. Na slici se mogu uočiti pet osnovnih scenarija: *web browser* ka *web* aplikaciji, SPA, *native* aplikacija ka *web* API-u, *web* aplikacija ka *web* API-u, *daemon* ili serverska apli-kacija ka *web* API-u. Azure AD omogućava klijentu i da veoma lako kombinuje više prostih pa da tako dobije složeniji scenarijo koji zadovoljava njegove potrebe.

Aplikacije mogu koristiti *Azure* AD radi obavljanja autentifikacije i autorizacije pomoću nekog od široko prihvaćenih i standardizovanih protokola za autentifikaciju i autorizaciju, a koje *Azure* AD podržava. Podržani protokoli od strane *Azure* AD-a su: OAuth 2.0, OpenID *Connect*, SAML 2.0 i WS-Federation (*Web Services Federation*). *Microsoft* za čitav niz programskih jezika i platformi nudi klijentima kroz odgovarajuće biblioteke otvorenog koda implementacije svakog od tih protokola.

U slučaju da je autentifikacija i autorizacija entiteta uspešna, *Azure* AD klijentskoj aplikaciji u odgovoru dostavlja i token. Format tokena i njegov sadržaj zavise od korišćenog protokola i izvesnih podešavanja. Tako na

primer ID tokeni i *access token*-i imaju format JWT-a čija je struktura JWS. U tokenu se nalaze različite tvrdnje koje se mogu iskoristiti u različite svrhe. Tako se, na primer, mogu koristiti za proveru ispravnosti samog tokena, utvrđivanje autorizacije entiteta, prikazivanje informacija o tom entitetu itd.

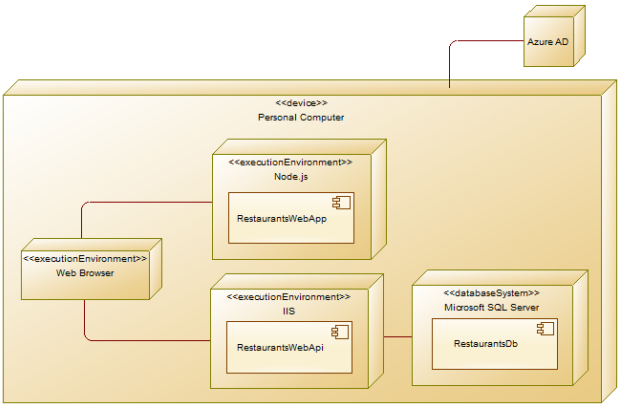



Slika 1. *Podržani scenariji i tipovi aplikacija od strane Azure AD-a*

## 3. MODEL SISTEMA ZA REZERVACIJU U RESTORANIMA

Sistem za rezervaciju u restoranima predstavlja softverski sistem koji kao osnovnu funkciju nudi onu koja omogućava gostima da kreiraju rezervacije stolova u restoranima. Sistem sprovodi autentifikaciju i autorizaciju korisnika koristeći servis *Azure* AD.

Na slici 2 prikazan je dijagram raspoređivanja koji se odnosi na sistem u slučaju njegovog razvoja i testiranja.




Slika 2. *Dijagram raspoređivanja sistema*

Kao što je već spomenuto u uvodu, sistem za rezervaciju u restoranima u svoj sastav uključuje dve komponente. Prva komponenta je nazvana *RestaurantsWebApp*. Ona je implementirana kao SPA i izvršava se na Node.js-u. Druga komponenta je nazvana *RestaurantsWebApi*. Ona je implementirana kao *web* API i izvršava se na IIS-u (*Internet Information Services*). *Microsoft* SQL *Server* predstavlja sistem za upravljanje bazama podataka na kojem je kreirana baza podataka koja je nazvana *RestaurantsDb*. Ona je formirana u skladu sa odgovarajućim konceptualnim modelom podataka. Između IIS-a i *Microsoft* SQL Server-a postoji komunikacija jer komponenta *RestaurantsWebApi* koristi bazu podataka *RestaurantsDb* u ostvarivanju svoje poslovne logike. Klijenti mogu pomoću nekog *Web Browser*-a da pristupe komponenti *RestaurantsWebApp*. *RestaurantsWebApp* koristi usluge *RestaurantsWebApi*-a preko poziva upućenih sa svog *frontend*-a (prezentacioni sloj) koji je učitan u *Web Browser*-u. *Web Browser*, *Node.js*, IIS i *Microsoft* SQL *Server* se izvršavaju na uređaju označenom sa *Personal Computer* (lični računar), koji predstavlja računar na kome se sistem razvija i testira. U obavljanju autentifikacije i autorizacije se koristi servis *Azure* AD pomoću OpenID *Connect* i OAuth 2.0. U odgovarajućem *Azure* AD *tenant*-u su registrovani *RestaurantsWebApp*, *RestaurantsWebApi* i korisnici. To sve znači da između *Personal Computer*-a i *Azure* AD-a postoji komunikacija preko interneta. Komunikacija sa servisom *Azure* AD, komponentom *RestaurantsWebApp* i komponentom *RestaurantsWebApi* se vrši korišćenjem protokola HTTPS. Time su ispoštovane preporuke koje su date u dokumentacijama za OAuth 2.0 i OpenID *Connect*.

## 4. IMPLEMENTACIJA SISTEMA ZA REZERVACIJU U RESTORANIMA

### 4.1. Registracija komponenti

Korišćenjem *web* aplikacije *Azure Portal* su obavljene sve neophodne aktivnosti nad servisom *Azure* AD kako bi ga sistem za rezervaciju u restoranima mogao koristiti radi obavljanja autentifikacije i autorizacije korisnika.

Za potrebe sistema je kreiran novi *Azure* AD *tenant*. U njemu su registrovane komponente *RestaurantsWebApi* i *RestaurantsWebApp*.

Obe komponente su registrovane kao *single-tenant* aplikacije. Njima je dozvoljeno da koriste protokol OAuth 2.0 zasnovanog na implicitnom *authorization grant*-u. Za obe komponente je specificirano da one prepoznaju, tj. definišu, tri uloge: *administrator*, *manager* i *guest*. Komponenti *RestaurantsWebApp* je omogućeno da komponenti *RestaurantsWebApi* šalje zahteve u ime korisnika koji je trenutno na nju prijavljen.

### 4.2. Implementacija RestaurantsWebApi-a

Komponenta *RestaurantsWebApi* je implementirana kao poseban projekat korišćenjem programskog jezika C# i *framework*-a (radni okvir) ASP.NET Web API 2.

Da bi se mogle koristiti mogućnosti koje pružaju OWIN i Katana, u projekat je dodat *NuGet* paket *Microsoft.Owin.Host.SystemWeb* čija je verzija 3.1.0. Pošto Katana obezbeđuje OWIN komponentu koja u protočnoj obradi HTTP zahteva može vršiti validaciju pristiglog *access token*-a izdatog od strane servisa *Azure* AD, u projekat je dodat i NuGet paket *Microsoft.Owin.Security.ActiveDirectory* čija je verzija 3.1.0.

Za registraciju OWIN komponente koja je sposobna da u protočnoj obradi vrši validaciju *access token*-a izdatog od *Azure* AD je iskorišćena *extension* metoda *UseWindowsAzureActiveDirectoryBearerAuthentication* koja je interfejsu *IAppBuilder* obezbeđena pomoću klase čiji je naziv *WindowsAzureActiveDirectoryBearerAuthenticationExtensions*.

Zaštita od neovlašćenog pristupa pojedinim akcijama kontrolera je ostvarena korišćenjem atributa *Authorize*, atributa koji je definisan pomoću klase *AuthorizeAttribute*.

Informacije o korisniku u čije ime se izvršava neka metoda se mogu dobiti pristupanjem statičkom svojstvu *Current* klase *ClaimsPrincipal*. To je iskorišćeno da se dobije korisničko ime korisnika koji je pokrenuo akciju.

### 4.3. Implementacija RestaurantsWebApp-a

Implementacija *frontend* dela komponente *RestaurantsWebApp* je ostvarena korišćenjem JavaScript *framework*-a AngularJS. AngularJS je obezbeđen uz pomoć *Bower* paketa *angular 1.7.5*. Radi ostvarivanja autentifikacije i autorizacije korisnika pomoću *Azure* AD-a je korišćena *Microsoft*-ova biblioteka ADAL JS (*Active Directory Authentication Library for JavaScript*) koja je obezbeđena pomoću *Bower* paketa *adal-angular 1.0.16*.

Paketom *adal-angular* su obezbeđena dva JavaScript fajla: *adal.js* i *adal-angular.js*. Fajl *adal.js* sadrži implementaciju elemenata i mehanizma koji omogućavaju ostvarivanje autentifikacije i autorizacije korisnika pomoću *Azure* AD-a korišćenjem protokola *OAuth* 2.0 zasnovanog na implicitnom *authorization grant*-u kroz protokol OpenID *Connect*. Ta implementacija ne zavisi ni od jedne JavaScript biblioteke, te se stoga može koristiti u *frontend*-u implementiranog (ne)korišćenjem nekog JavaScript *framework*-a. U fajlu *adal-angular.js* se nalazi implementacija koja se ponaša kao AngularJS omotač oko stvari obezbeđenih kroz *adal.js*. Ona u AngularJS aplikacijama olakšava korišćenje mogućnosti koje pruža *adal.js*. Zbog svega toga su iskorišćena oba fajla.

Prilikom razvoja sistema je uočeno da ADAL JS ima određeni nedostatak. Taj nedostatak se može ilustrovati na sledećem primeru. Neka je korisnik neprijavljen na sistem i neka želi da prikaže spisak restorana. Nakon pokretanja te akcije *frontend RestaurantsWebApp*-a će uputiti zahtev na putanju */api/restaurants RestaurantsWebApi*-a. Taj zahtev će uputiti koristeći metodu HTTP GET. Za tu putanju i tu HTTP metodu *RestaurantsWebApi* poseduje akciju koja omogućava obradu zahteva, tj. obezbeđuje dostavu spiska restorana. Pri tome, ta akcija ne zahteva da korisnik bude prijavljen. ADAL JS će presresti zahtev, a zatim će pokušati da dobavi odgovarajući token kako bi ga ubacio u zaglavlje zahteva jer je prilikom konfiguracije ADAL JS-a specificirano da se *RestaurantsWebApi* nalazi u spisku odredišta za koje je neophodan token, tj. u spisku odredišta zaštićenih pomoću Azure AD-a, i zato što se ta adresa ne nalazi u spisku izuzetaka, odnosno adresa za koje nije potreban token. Pošto u tome ne uspeva, ADAL JS će prekinuti isporuku HTTP zahteva. Ishod toga je da se neće uspeti dobaviti spisak restorana i da ga korisnik neće videti.

Za tu istu adresu, ali za metodu HTTP POST, *RestaurantsWebApi* poseduje akciju koja zahteva da korisnik bude prijavljen na sistem. Ona omogućava dodavanje novog restorana.

Da je ova adresa navedena u spisku izuzetaka i da se za prijavljenog korisnika upućuje zahtev kojim se želi dodati novi restoran, ADAL JS u zahtev ne bi ubacio odgovarajući token, pa bi *RestaurantsWebApi* odbila taj zahtev. Da bi se izbegle takve i slične situacije, implementacija ADAL JS-a je izmenjena tako da ADAL JS propušta svaki HTTP zahtev upućen nekom *web* API-u iako ne uspe da dobavi i u njega ubaci odgovarajući token, jer *web* API sam treba da odluči da li će ga prihvatiti ili odbiti. Izmena je načinjena u *adal-angular.js*, a izmenjeni segment je prikazan na stranici [7]. Sadržaj fajla *adal-angular.js* sa primenjenom izmenom je prikazan na stranici [8].

## 5. ZAKLJUČAK

U radu je opisano rešavanje problema autentifikacije i autorizacije korisnika korišćenjem mogućnosti koje pruža *Microsoft Azure*. Date su teorijske osnove vezane za *Microsoft Azure* kao jedan od dostupnih sistema koji omogućavaju *cloud computing*. Opisan je servis *Azure* AD, servis koji nudi *Microsoft Azure* i koji omogućava obavljanje autentifikacije i autorizacije.

Primena *Microsoft Azure*-a, tj. servisa *Azure* AD, je pokazana na primeru softverskog sistema za rezervaciju u restoranima. Prilikom razvoja sistema su korišćene već dostupne biblioteke koje omogućavaju korišćenje *Azure* AD-a radi ostvarivanja autentifikacije i autorizacije. Jedna od njih je ADAL JS. U njoj je uočen i otklonjen nedostatak.

Prilikom prijavljivanja na sistem korisnik unosi korisničko ime i lozinku. Nivo sigurnosti bi se mogao povećati, na primer, primenom multi-faktorske autentifikacije za koju postoji podrška od strane *Azure* AD-a.

## 6. LITERATURA


[1] *"Cloud Computing Synopsis and Recommendations"*, https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-146.pdf

[2] Microsoft Azure, *"What is cloud computing?"*, https://azure.microsoft.com/en-us/overview/what-is-cloud-computing/

[3] Microsoft Azure, *"What is Azure?"*, https://azure.microsoft.com/en-us/overview/what-is-azure/

[4] OWASP, *"Authentication Cheat Sheet"*, https://www.owasp.org/index.php/Authentication_Cheat_Sheet

[5] OWASP, *"Access Control Cheat Sheet"*, https://www.owasp.org/index.php/Access_Control_Cheat_Sheet

[6] Microsoft Azure, *"What is Azure Active Directory?"*, https://docs.microsoft.com/en-us/azure/active-directory/fundamentals/active-directory-whatis

[7] https://github.com/AzureAD/azure-activedirectory-library-for-js/pull/688/commits/db2dde58c6a344f446efffc7baadf04ffaed5af8

[8] https://github.com/AzureAD/azure-activedirectory-library-for-js/blob/db2dde58c6a344f446efffc7-baadf04ffaed5af8/lib/adal-angular.js


### Kratka biografija:


**Dražen Đanić** rođen je 1993. godine u Vrbasu. Srednju tehničku školu „Mihajlo Pupin" u Kuli završio je 2012. god., pri tome dobio diplomu „Vuk Karadžić". Iste godine upisao se na osnovne akademske studije na Fakultetu tehničkih nauka u Novom Sadu, odsek Računarstvo i automatika. Osnovne akademske studije je uspešno završio 2016. odbranom diplomskog rada na temu „Podsistem za generisanje sertifikata u okviru informacionog sistema skupštine grada". Iste godine je upisao master akademske studije na istom fakultetu i smeru.