

Tehnike digitalnog potpisa u primjeni potpisivanja pisanih provjera znanja

Starčević, Simona

Undergraduate thesis / Završni rad

2020

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Bjelovar University of Applied Sciences / Veleučilište u Bjelovaru**

Permanent link / Trajna poveznica: <https://um.nsk.hr/um:nbn:hr:144:592711>

Rights / Prava: [In copyright](#)/[Zaštićeno autorskim pravom.](#)

Download date / Datum preuzimanja: **2024-12-21**



Repository / Repozitorij:

[Repository of Bjelovar University of Applied Sciences - Institutional Repository](#)



VELEUČILIŠTE U BJELOVARU
PREDDIPLOMSKI STRUČNI STUDIJ RAČUNARSTVO

**TEHNIKE DIGITALNOG POTPISA U PRIMJENI
POTPISIVANJA PISANIH PROVJERA ZNANJA**

Završni rad br. 03/RAČ/2020

Simona Starčević

Bjelovar, listopad 2020.



Veleučilište u Bjelovaru

Trg E. Kvaternika 4, Bjelovar

1. DEFINIRANJE TEME ZAVRŠNOG RADA I POVJERENSTVA

Kandidat: **Starčević Simona**

Datum: 31.08.2020.

Matični broj: 002100

JMBAG: 0319002065

Kolegij: **SIGURNOST RAČUNALA I PODATAKA**

Naslov rada (tema): **Tehnike digitalnog potpisa u primjeni potpisivanja pisanih provjera znanja**

Područje: **Tehničke znanosti**

Polje: **Računarstvo**

Grana: **Informacijski sustavi**

Mentor: **Dario Vidić, mag.ing.el.techn.inf.**

zvanje: **predavač**

Članovi Povjerenstva za ocjenjivanje i obranu završnog rada:

1. Tomislav Adamović, mag.ing.el., predsjednik
2. Dario Vidić, mag.ing.el.techn.inf., mentor
3. Ivan Sekovanić, mag.ing.inf.et comm.techn., član

2. ZADATAK ZAVRŠNOG RADA BROJ: 03/RAČ/2020

U radu je potrebno korištenjem SQLite baze podataka, Angulara za Web aplikaciju te .NET Framework (C#) REST API-a izraditi sustav za generiranje i dohvaćanje digitalnog potpisa te potpisivanje pisane provjere znanja.

Objediniti segmente digitalnog potpisa: autentičnost, autorizacija i verifikacija.

Omogućiti studentima kreiranje ključa za potpisivanje kolokvija i potvrđivanje identiteta.

Zadatak uručen: 31.08.2020.

Mentor: **Dario Vidić, mag.ing.el.techn.inf.**



Zahvala

Zahvaljujem se mentoru Dariu Vidiću, magistru inženjeru elektrotehnike i informacijske tehnologije na vremenu i strpljenju za sva moja pitanja te zahvaljujem na svim sugestijama i savjetima tijekom pisanja završnog rada.

Veliko hvala mom radnom kolektivu Axess Cro na pruženom znanju i mogućnosti učenja uz mentorstvo iznimnih stručnjaka.

SADRŽAJ

1. UVOD	1
2. DIGITALNO UČENJE (E - UČENJE)	2
3. KRIPTOGRAFIJA	4
3.1 <i>Simetrični kriptosustavi</i>	4
3.2 <i>Asimetrični kriptosustavi</i>	5
3.3 <i>Hibridni kriptosustavi</i>	6
4. DIGITALNI POTPIS	7
4.1 <i>Hash funkcija</i>	9
5. ALGORITMI ZA DIGITALNO POTPISIVANJE	11
6. KORIŠTENE TEHNOLOGIJE	13
6.1 <i>REST API</i>	13
6.2 <i>Baza podataka</i>	19
6.2.1 <i>Dizajn baze podataka</i>	20
6.3 <i>AngularJS</i>	21
6.4 <i>Zahjevi na sustav</i>	24
7. FUNKCIONALNOST APLIKACIJE I KORISNIČKO SUČELJE	25
7.1 <i>Administrator</i>	26
7.2 <i>Profesor</i>	27
7.3 <i>Student</i>	29
8. ZAKLJUČAK	32
9. LITERATURA	33
10. OZNAKE I KRATICE	34
11. SAŽETAK	35
12. ABSTRACT	36
13. PRILOZI	37

1. UVOD

Zahvaljujući današnjoj digitalizaciji i tehnološkom napretku obrazovanje nikada nije bilo dostupnije. Učenje u obrazovnim ustanovama, knjižnicama i vlastitoj kući preuzima digitalni oblik učenja. Ocjenjivanje se izvodi pisanim provjerama, seminarima i usmenim ispitivanjima putem video komunikacijskih platforma. Usprkos poznavanja sudionika prilikom usmenog ispitivanja, ono se gubi kada je riječ o pisanom obliku ispita. Iz tog razloga izrađena je aplikacija koja na siguran i vjerodostojan način osigurava neometan oblik digitalnog obrazovanja. Važno je razumjeti kriptografske metode i oblike zaštite podataka pomoću šifriranja odgovarajućim algoritmima. U idućim poglavljima detaljno se objašnjene vrste kriptografije, algoritmi za šifriranje podataka i funkcionalnost same aplikacije čiji je cilj zaštita podataka studenata i nastavnog osoblja.

Kako bi korisnik koristio aplikaciju, prilikom povezivanja na nju mora se prijaviti s korisničkim podacima te je pristup privatnim mrežama poput sustava za upravljanje učenjem (engl. „*Learning management system*“ - LMS) dodatno osiguran procesom autorizacije i autentifikacije. Objašnjene su razine pristupa prijavljenih korisnika aplikacije i vidljivost određenih podataka.

Autorizacija je proces kojim se autentificiranom korisniku dodjeljuju ili oduzimaju prava pristupa. Autoriziranom korisniku (studentu, profesoru ili administratoru) istaknute su određene funkcionalnosti s obzirom na razinu prava. Povjerljivost podataka jedna je od najvažnijih problematika koja se u radu postiže RSA algoritmom s ciljem kriptiranja podataka.

Fakultetu se omogućuje provjera identiteta korisnika i tako se sprječava neželjen upad. Kod utvrđivanja autentičnosti seminarskih radova za potrebe javnih visokih učilišta Republike Hrvatske koriste se Turnitin i PlagScan. [1] Upravo je tako omogućeno profesorima, ali i studentima provjera radova od mogućih plagijata.

2. DIGITALNO UČENJE (E - UČENJE)

Novonastala epidemiološka situacija otkrila je, ali i nametnula digitalni oblik učenja na daljinu. Obrazovne ustanove pokušavaju uspostaviti jednako kvalitetan oblik obrazovanja pazeći na dosljednost i kvalitetu samog nastavnog programa. Brzi prelazak na digitalni oblik učenja prisilio je mnoge na radikalne promjene. Obrazovanje u današnjem obliku bilo je standardno prihvaćeno uz nepripremljenost na digitalni oblik nastave. Mnoga sveučilišta i škole okrenula su se digitalizaciji obrazovanja. Predavanja preko raznih video komunikacijskih platformi poput Zoom-a i Skype-a postala su svakodnevica. Prednost takvog modela učenja je povećan broj materijala za učenje te mogućnost preslušavanja video materijala. Na taj način studentima i učenicima je omogućeno više puta proći nastavno gradivo te tako smanjiti poteškoće u učenju.

Vjeruje se u autentičnost samih ispita te u autorizaciju studenata. Sigurnost aplikacije je iznimno važna te se smatra najvećim izazovom završnog rada. Nedostaci takvog modela obrazovanja su oduzeta interakcija, pouzdana internetska veza, nedostatak računala i mnogi drugi. Sve se češće podržava digitalni oblik nastave u svrhu njegovog napretka. Mogućnost digitalne nastave od kuće, ili učenje putem digitalnih platformi (npr. Youtube i razni edukativni video zapisi) bilježe jednaku razinu savladavanja gradiva. Brojni članci i velik broj knjiga u digitalnom obliku uvijek su na raspolaganju studentima i učenicima. Izrađena aplikacija koja će detaljnije biti objašnjena u idućim poglavljima zamišljena je s ciljem nesmetanih pisanja ispita.

Ljudski um najbrže i najučinkovitije pamti informacije u video i audio obliku. Mogućnost višestrukog ponavljanja i veća moć usvajanja gradiva glavni su kriteriji za budući razvoj digitalnog oblika učenja. Do sada je obrazovanje bilo digitalizirano na razini e-Dnevnika i raznih sustava za upravljanje učenjem (engl. *LMS*) poput Moodle-a, Blackboard-a, Zoom-a i ostalih. Najvažniji problemi prilikom online izvođenja pisanja ispita prvenstveno su prepisivanje i plagiranje. [2] Učenici i studenti pozivaju se na pridržavanje akademske čestitosti te su obaviješteni o mogućnih sankcijama. Zbog neodgovarajuće kibernetičke ustanove mnoge obrazovne ustanove izložene su visokom riziku kibernetičkog napada. Curenje podataka i povrede privatnosti rezultat je iskorištavanja propusta Zooma od strane zlonamjernih napadača.

Umjesto pisanja bilješki što dovodi do smanjene koncentracije, digitalne platforme omogućuju usmjeravanja pažnje na samo gradivo bez opterećenja s bilješkama. Glavni razlog je sve veća prisutnost digitalnih uređaja i njihovo često korištenje. Brojne tvrtke i obrti “preko noći” su se digitalizirali kako ne bi snosili posljedice nastale krize, a koje se neminovno može prelići i na opće gospodarsko stanje. Valja se okrenuti inovativnim oblicima ne dozvoljavajući ovakvim pojavama poput pandemije zaustavljanje obrazovnih procesa. Nastavno osoblje dužno je prilagoditi nastavu novonastaloj epidemiološkoj situaciji te prilagoditi pedagoške mjere u svrhu nesmetane provedbe nastave. U nastavku su prikazane funkcionalnosti aplikacije te kako se pomoću nje osigurava sigurnost pisanih provjera i kasniji pregled ispita.

3. KRIPTOGRAFIJA

Svakodnevni oblici digitalne komunikacije postali su sve zastupljeniji isto kao i potrebna za njenom sigurnošću. Stari Grci su u 5.stoljeću prije Krista upotrebljavali razne elemente kriptiranja, koje je s grčkog jezika doslovno prevedena kao tajnopis. Kriptografija je znanstvena disciplina koja se bavi proučavanjem metoda za slanje poruka u takvom obliku da ih samo onaj kome su namijenjene može pročitati.[4] Kriptografija omogućuje komunikaciju dviju stranki preko nesigurnog komunikacijskog kanala, a treća stranka ne može razumjeti njihove poruke.[5] Osnovni kriptografski pojmovi su: šifriranje (kodiranje), dešifriranje (dekodiranje) i ključ. [6] Pošiljalatelj šifrira podatke te ih šalje primatelju dok treća osoba pokušava dešifrirati podatke bez ključa. Primatelj prima podatke od pošiljalatelja, dešifrira ih te čita. Kako bi algoritam bio ispravan potrebno je da pošiljalatelj i primatelj imaju isti ključ.

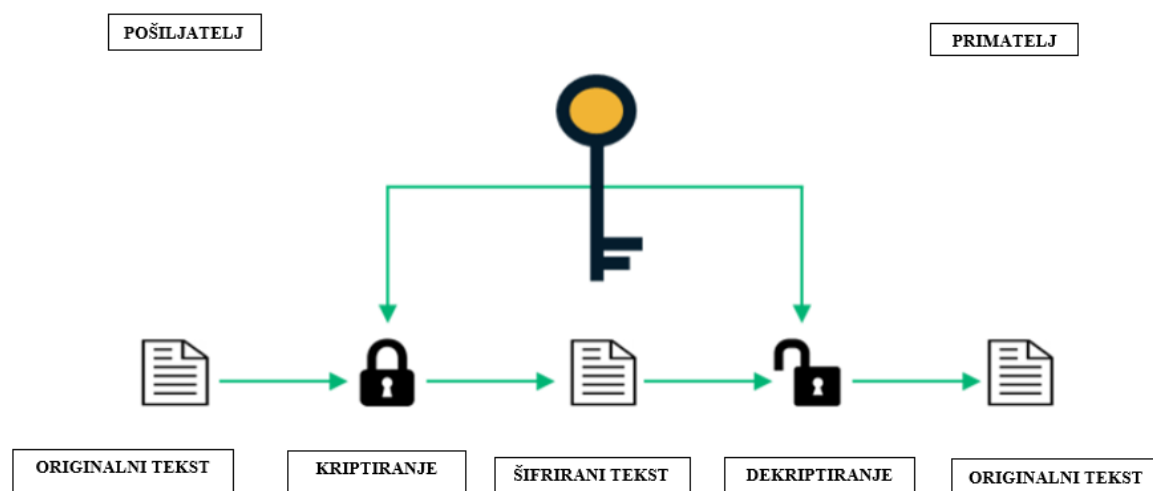
Kriptosustave možemo razvrstati na osnovu tajnosti i javnosti ključa na simetričnu, asimetričnu i hibridnu kriptografiju

3.1 *Simetrični kriptosustavi*

Osnovna osobina simetričnih kriptosustava je korištenje samo jednog ključa, odnosno jednak ključ za kriptiranje i dekriptiranje (slika 1.). Dok god je simetrični ključ kao tajan zadržan između dva korisnika, komunikacija je sigurna.[7] Najveći problem je nezaštićenost komunikacijskog kanala te treća osoba može presresti poruku. Treća osoba može pročitati poruku samo ako dođe u posjed ključa kojeg je pošiljalatelj prvotno koristio. Glavni problem simetrične kriptografije je nemogućnost digitalnog potpisivanja. [5] Sigurnost se postiže osiguravanjem tajnosti ključa te se još koristi naziv i *kriptosustavi s tajnim ključem*. Općenito su simetrični kriptosustavi mnogo brži nego asimetrični.

Najpoznatiji simetrični kriptografski algoritmi:

- DES (engl. *Data Encryption Standard*)
- AES (engl. *Advanced Encryption Standard*)
- 3DES (engl. *Triple Data Encryption Standard*)



Slika 1. Model simetrične kriptografije

3.2 Asimetrični kriptosustavi

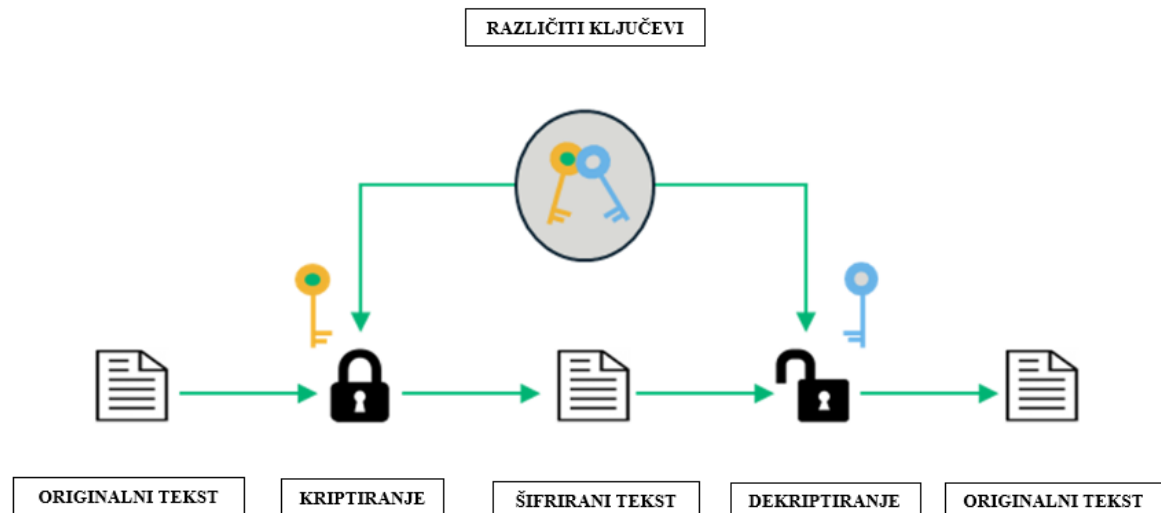
Asimetrična kriptografija, odnosno *kriptosustavi s javnim ključem* osmišljeni su tako da se ključ za dekriptiranje ne može izračunati iz ključa za kriptiranje u prihvatljivom vremenu. Razmjena tajnih ključeva moguća preko nesigurnih komunikacijskih kanala. Koristi se par povezanih ključeva: javni i privatni ključ. Javni ključ koji je dostupan svima služi za šifriranje originalnog teksta prije slanja (vidljivo na slici 2.). Da bi se poruka dekriptirala i kako bismo je mogli pročitati, trebamo imati privatni ključ. Javni i privatni ključevi su matematički povezani, ali iz njih se ne može saznati privatni ključ. Vrijeme potrebno za izračunavanje tajnog ključa iz poznatog javnog ključa mjeri se u milijunima godina na danas najjačim raspoloživim računalima. U asimetričnoj kriptografiji sigurnost je moguće očuvati tako da jedan od dva ključa mora ostati tajan.[5]

Glavni nedostatak je matematička složenost asimetričnih algoritama u odnosu na simetrične algoritme te se zbog toga ne koriste za kriptiranje većih količina podataka.

Osim klasičnog kriptiranja, kriptografija javnog ključa koristi se i za digitalne potpise. Autentičnost pošiljatelja koji potpisuje poruku privatnim tajnim ključem može biti provjerena pomoću javnog ključa.

Najpoznatiji simetrični kriptografski algoritmi:

- RSA (Rivest-Shamir-Adleman) algoritam
- DSA (engl. *Digital Signature Algorithm*)
- ECDSA (engl. *Elliptic Curve Digital Signature Algorithm*)



Slika 2. Model asimetrične kriptografije

3.3 Hibridni kriptosustavi

Hibridni kriptosustav razvijen je zbog raznih prednosti i nedostataka simetričnih i asimetričnih kriptosustava. Simetrični sustavi imaju nedostatak nesigurnih izmjena ključa između pošiljatelja i primatelja, a asimetrični sustavi nisu pogodni za izmjenu velike količine podataka. Upravo iz tog razloga nastala je ideja kombiniranja oba sustava kako bi se zaobišli nedostaci prethodnih kriptosustava. Prvo se asimetričnom kriptografijom razmjenjuje tajni ključ za ostvarivanje simetrične kriptografije, koja se kasnije koristi za prijenos velike količine podataka. [8] Zahvaljujući svojim prednostima hibridni sustavi su postali najrašireniji kriptosustavi.

Najpoznatiji hibridni kriptografski algoritmi:

- SSL (engl. *Secure Sockets Layer*)
- TLS (engl. *Transport Layer Security*)
- PGP (engl. *Pretty Good Privacy*)

4. DIGITALNI POTPIS

U fizičkom svijetu uobičajeno je koristiti potpis kao način zaštite i vjerodostojnosti informacije. Slično tome, digitalni potpis je tehnika koja povezuje osobu i digitalne podatke. Digitalni potpis (engl. *digital signature* - DS) definiran je kao “šifriranje kojim se dokazuje autorstvo”, odnosno digitalna verzija vlastoručnog potpisa koji zakonski vrijedi isto kao i rukom potpisan dokument. Digitalni potpis smatra se numeričkom vrijednošću predstavljena nizom znakova. Kreiranje digitalnog potpisa složen je matematički postupak koji je moguće kreirati jedino korištenjem računala. Tehnologija digitalnog potpisa koristi tehniku asimetričnog kriptiranja.[9]

Digitalni potpis je tehnika koja veže osobu za digitalne podatke. Kriptografija digitalnog potpisa moguća je pomoću privatnog i javnog ključa primijenjenih u izradi aplikacije. Privatni ključ koristi se za šifriranje i vlasnik ključa dužan je brinuti za sigurnost ključa. Javni ključ je, vidljivo iz imena, javan.

Samo se pomoću javnog ključa mogu dešifrirati podaci šifrirani privatnim ključem kako bi se saznali izvorni podaci. Ukratko, za digitalno potpisivanje dokumenta pošiljatelj koristi svoj privatni ključ. Da bi primatelj mogao ovjeriti digitalni potpis, mora koristiti javni ključ pošiljatelja. Pretpostavimo da student želi profesoru putem kreirane aplikacije poslati dokument te pritom zadržati autentičnost kolokvija :

- 1) Student će odabrati kolokvij koji želi poslati profesoru (*Profile* → *Documents*) te pritisnuti “Potpiši”.
- 2) Poslužitelj (REST.API) će izračunati jedinstvenu hash vrijednost koju će dodijeliti potpisanom dokumentu.
- 3) Vrijednost hash-a šifrirat će se pomoću studentovog privatnog ključa kojem pristup ima jedino student te će dokument biti spreman za prijenos.

- 4) Kada profesor zaprimi dokument, aplikacija će primijetiti da je dokument potpisan. Zatim će nastaviti s dešifriranjem digitalnog potpisa pomoću studentovog javnog ključa. Ako se dešifrirani javni ključ ne podudara s hash vrijednošću, dokument je nevažeći.

Digitalni potpis mora zadržati svoje temeljne karakteristike: integritet, autentičnost i nepobitnost.

- Integritet

Profesor može potvrditi da studentova poruka nije promijenjena na putu do primatelja (profesora). Svaka izmjena u poruci rezultirat će potpuno drugačijim potpisom.

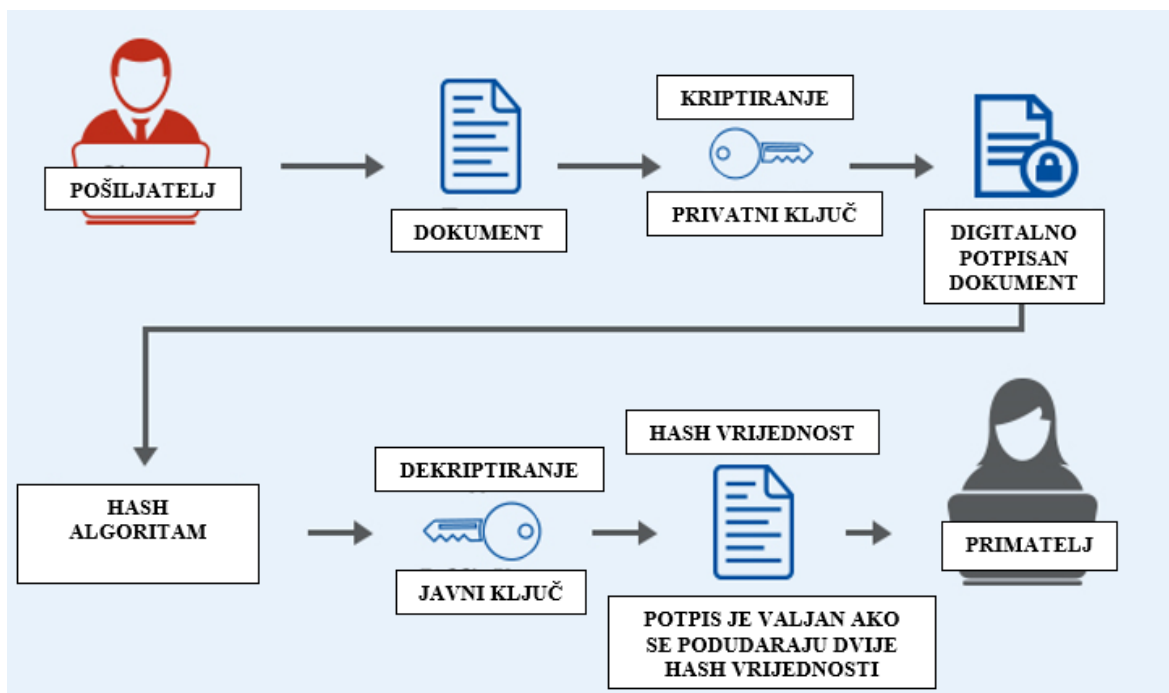
- Autentičnost

Sve dok se studentov privatni ključ čuva u tajnosti, profesor svojim javnim ključem može potvrditi da je digitalne potpise stvorio samo određeni student.

- Nepovratnost

Jednom kada se potpis generira, student neće moći poreći da je potpisao dokument osim u slučaju ugroženosti studentova privatnoga ključa. Isto vrijedi i za kartice bankovnih računa. Ako se plaća s karticom koja ne traži pin ona se smatra važećom sve dok vlasnik kartice ne prijavi, primjerice, njezin nestanak.

Važno je napomenuti da su digitalni potpisi izravno povezani sa sadržajem poruke. Za razliku od rukom potpisanih potpisa koji imaju tendenciju biti jednaki bez obzira na poruku, svaka digitalno potpisana poruka imat će drugačiji digitalni potpis. Kad primatelj (u našem slučaju profesor) primi poruku, može provjeriti valjanost digitalnog potpisa pomoću javnog ključa koji je osigurao pošiljatelj (student). Tako profesor može biti siguran da je student koji je priložio kolokvij i potpisao kolokvij jer samo student ima ključ koji odgovara tom javnom ključu. Presudno je da student drži svoj privatni ključ u tajnosti jer u slučaju da druga osoba sazna privatni ključ studenta može se pretvarati da potpisuje u ime otuđenog studenta.



Slika 3. Prikaz tijeka digitalnog potpisivanja

4.1 Hash funkcija

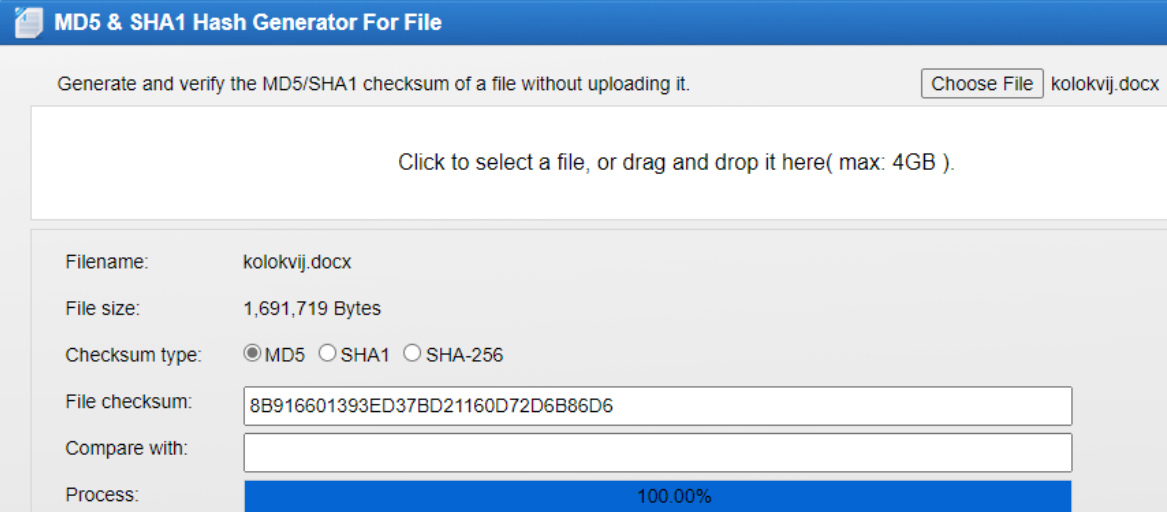
U kriptografiji se pojavljuju takozvane kriptografske hash funkcije te se mogu koristiti za generiranje hash vrijednosti koje djeluju kao jedinstveni digitalni otisak prsta. To znači da bi svaka promjena ulaznih podataka (poruka) rezultirala potpuno drugačijim izlazom (hash vrijednost). Ulaz hash funkcije je niz znakova proizvoljne duljine, a izlaz je niz znakova fiksne duljine (npr. 128 ili 160 bitova). Promjena jednog bita ulaza rezultira potpuno drugačijim izlazom. [5] Upravo je to razlog zašto se kriptografske hash funkcije koriste za provjeru autentičnosti digitalnih podataka.

Nakon što se podaci hashiraju, pošiljatelj poruke mora ih potpisati privatnim ključem, a primatelj poruke može provjeriti valjanost poruke pomoću odgovarajućeg javnog ključa koji osigurava pošiljatelj (slika 3.). Ako privatni ključ nije uključen kad se potpis generira, primatelj poruke neće moći iskoristiti odgovarajući javni ključ za provjeru njegove vrijednosti. Javni i privatni ključ generira pošiljatelj poruke, a primatelju se dijeli samo javni ključ. Hash funkcije se primjenjuju za digitalne potpise, hash-tablice, za detekciju kopija binarnih sadržaja itd.

U nastavku su prikazani hash algoritmi za digitalni potpis:

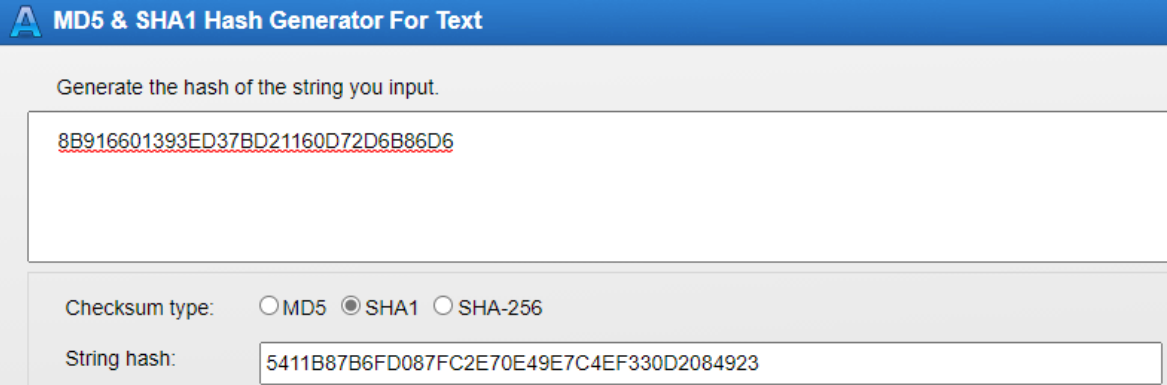
- MD5 (engl. *Message Digest Algorithm 5*)
- SHA-1, SHA-2, SHA-3 (engl. *Secure Hash Algorithm*)

Ukoliko želimo priložiti dokument možemo saznati hash vrijednost MD5 algoritma vidljivog na slici 4. Na slici 5 vidimo pretvorbu vrijednosti MD5 algoritma u SHA-1 algoritam. U slučaju da dokument ima barem jednu izmjenu, hash vrijednost određenog algoritma dobiva posve drugačiju vrijednost.



The screenshot shows a web interface titled "MD5 & SHA1 Hash Generator For File". It includes a "Choose File" button next to the filename "kolokvij.docx". Below the file selection area, there is a text box containing the MD5 hash: "8B916601393ED37BD21160D72D6B86D6". The "Checksum type" is set to "MD5" (selected with a radio button). A progress bar at the bottom indicates "Process: 100.00%".

Slika 4. Prikaz vrijednosti MD5 hash algoritma



The screenshot shows a web interface titled "MD5 & SHA1 Hash Generator For Text". It includes a text input field containing the MD5 hash "8B916601393ED37BD21160D72D6B86D6". Below the input field, the "Checksum type" is set to "SHA1" (selected with a radio button). A text box at the bottom displays the resulting SHA-1 hash: "5411B87B6FD087FC2E70E49E7C4EF330D2084923".

Slika 5. Dobivena vrijednost SHA-1 algoritma

5. ALGORITMI ZA DIGITALNO POTPISIVANJE

U nastavku su opisana tri najčešća algoritma korištena za digitalno potpisivanje koji se temelje na kriptografiji javnog ključa, odnosno algoritmima asimetrične kriptografije:

- DSA (engl. *Digital Signature Algorithm*)

Algoritam digitalnog potpisa (DSA) odnosi se na standard za digitalne potpise. Uveo ga je 1991.godine Nacionalni institut za standarde i tehnologiju (NIST) kao bolju metodu stvaranja digitalnih potpisa. Uz RSA, DSA se smatra jednim od najpoželjnijih algoritama digitalnog potpisa koji se danas koristi. DSA koristi jedinstvene matematičke funkcije za stvaranje digitalnog potpisa. Algoritam digitalnog potpisa koristi javni ključ za autentifikaciju potpisa, ali je postupak autentifikacije složeniji u usporedbi s RSA.

- ECDSA (engl. *Elliptic Curve Digital Signature Algorithm*)

Algoritam digitalnog potpisa eliptičke krivulje (ECDSA) algoritam je digitalnog potpisa (DSA) koji koristi ključeve izvedene iz kriptografije eliptične krivulje (ECC). Funkcionalno pruža jednak ishod kao i drugi algoritmi digitalnog potpisivanja, jer se ECDSA temelji na učinkovitijoj kriptografiji eliptičke krivulje. ECDSA zahtijeva manje ključeve za pružanje jednake sigurnosti i stoga je učinkovitiji. [9]

- RSA (Rivest – Shamir – Adleman)

RSA je najrašireniji i najpopularniji asimetrični sustav nazvan po prvim slovima prezimena njegovih autora (Ron Rivest, Adi Shamir i Len Adleman) nastao je 1977 godine. Proces generiranja ključeva algoritmom RSA danas ga čini sigurnim i pouzdanim jer sadrži visoku razinu složenosti u usporedbi s drugim kriptografskim algoritmima. RSA algoritam se može koristiti za potpisivanje poruke, tako da osoba A može potpisati poruku svojim privatnim ključem. Osoba B može potvrditi poruku pomoću javnog ključa osobe A.

Duljina ključa (bit)		Vrijeme (sekunda)	
ECDSA	RSA	ECDSA	RSA
163	1024	0.08	0.16
233	2240	0.18	7.47
283	3072	0.27	9.80
409	7680	0.64	133.90
571	15360	1.44	679.06

Tablica 1. Performanse RSA i ECDSA algoritma

Završni rad temelji se na RSA algoritmu. DSA i ECDSA su možda brži algoritmi asimetrične kriptografije za generiranje potpisa, ali su sporiji za provjeru valjanosti i za kriptiranje iste duljine ključa (tablica 1).

Za razliku od RSA algoritma, DSA i ECDSA algoritmi brže generiraju potpise, ali su puno sporiji kada je u pitanju provjera potpisa iste duljine ključa (tablica 1.). Zbog navedenih činjenica, koristi se RSA algoritam prilikom digitalnog potpisivanja.

6. KORIŠTENE TEHNOLOGIJE

Prije kreiranja aplikacije i njezine namjene bilo je važno odabrati platforme prilikom izrade aplikacije za primjenu digitalnih potpisa u svrhu ocjenjivanja pisanih provjera znanja. Kako bi aplikacija bila funkcionalna potrebni su joj strana poslužitelja (engl. *backend*), baza i stranica za klijente (engl. *frontend*).

REST API je korišten za *backend* aplikacije zbog potrebe za URL adresom i standardnim HTTP metodama: GET, POST, PUT i DELETE koje su potrebne za nesmetani rad te kompleksnosti aplikacije. Nakon odabira određene metode REST API-a svaka metoda izvršava svoje zadatke te je rezultat akcija prikazan u URL adresi kako bi ga i krajnji korisnik mogao razumjeti.

Programski jezici korišteni u AngularJS-u su HTML, SCSS i JavaScript. Za skup alata za razvoj (engl. *framework*) je odabran AngularJS zbog izvrsne povezanosti s *backend*-om te jednostavne upotrebe za izradu *frontend*-a web aplikacije koji predstavlja vizualizaciju same aplikacije.

Baza podataka je most između frontenda i backenda, tj. komunikacijski kanal. U bazi su pohranjeni podaci korišteni u aplikaciji te pomoću HTTP metoda moguće je obrisati ili izmijeniti podatke u SQLite bazi podataka. SQLite baza koristi jedinstvenu SQL sintaksu te je zato jednostavna za korištenje i ne razlikuje se od drugih baza podataka kada su u pitanju naredbenih riječi koje su ujedno i upiti. SQLite baza je namijenjena za jednostavnije aplikacije kao u slučaju ovog završnog rada.

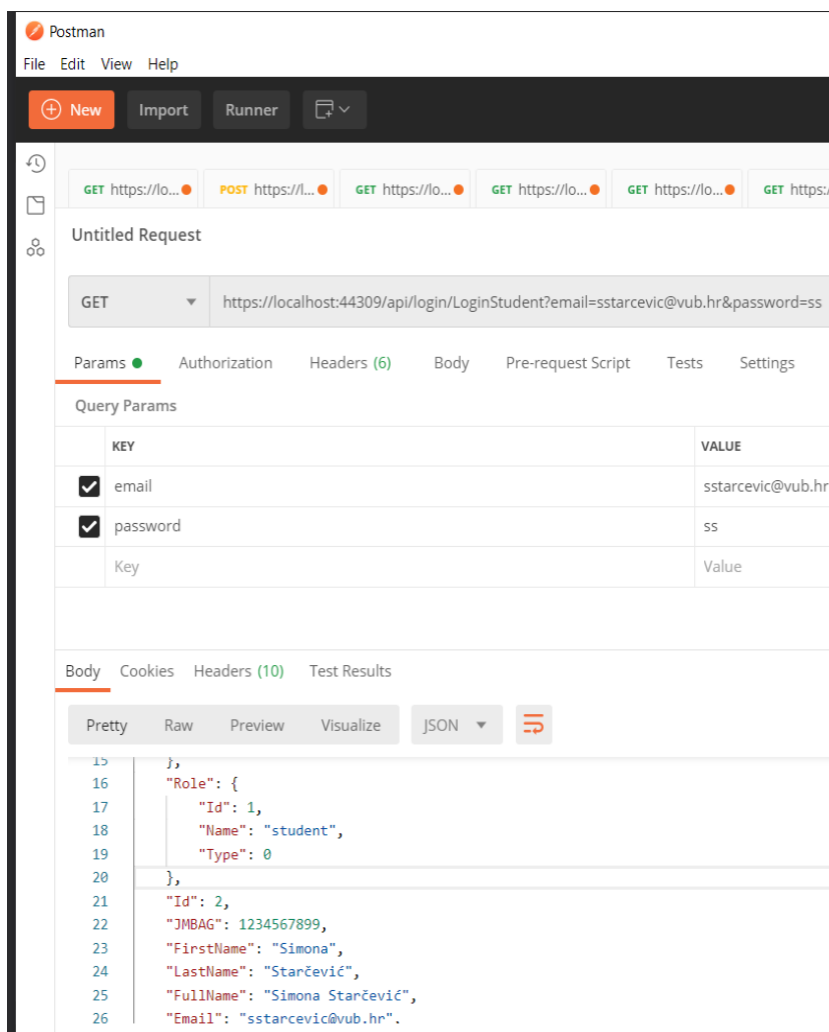
Korištene tehnologije odabrane su na temelju iskustva stečenog u radu s navedenim tehnologijama tijekom odrađene studentske prakse.

6.1 REST API

Aplikacijsko programsko sučelje (engl. *application programming interface - API*) REST (engl. *Representational State Transfer*) daje vanjskim aplikacijama način za obavljanje upita i ažuriranja aplikacijskih podataka. Reprezentativni prijenos stanja, REST je reprezentacijsko stanje prijenosa, odnosno označava način komunikacije klijenta i poslužitelja. [10] REST API podrška koristi standardne HTTP metode:

- **GET** - dohvaćanje zapisa

GET je najpoznatija HTTP metoda koja zahtjeva prikaz određenog resursa. Šaljemo li podatke putem GET-a, podaci se šalju u URL-u i nakon što se stranica učita podaci su vidljivi korisniku.

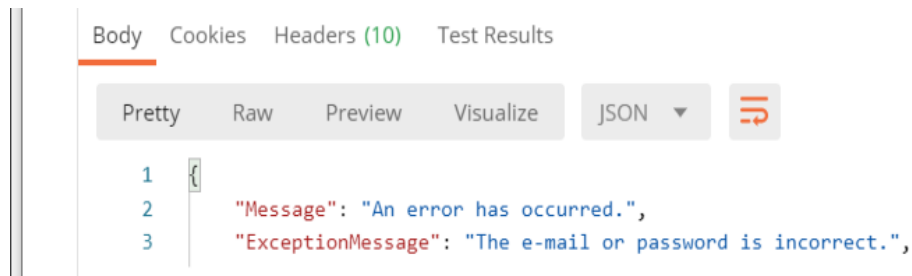


Slika 6. Prikaz HTTP GET metode pomoću Postman-a

Na slici 6. pomoću aplikacije Postman testiramo GET API metodu. Postman je jednostavno i pristupačno korisničko sučelje pomoću kojeg se šalju HTML zahtjevi (engl. *request*) bez nepotrebnog pisanja koda samo radi testiranja funkcionalnosti API-a. URL:

`https://localhost:44309/api/login/LoginStudent?email=sstarcevic@vub.hr&password=ss`

URL prikazuje dohvaćanje studenta sa *LoginControllera* na API-u dohvaćajući e-mail i lozinku (engl. *password*). Ako je upit (e-mail i lozinka) ispravan student se prijavljuje na aplikaciju te je potvrđena autentifikacija. U slučaju da upit nije valjan unutar Postman-a prikazuje se poruka o neispravnom e-mailom ili lozinci. Prikazuje se HTTP status kod *500 Internal Server Error*. Server označuje da je poslužitelj naišao na neočekivano stanje koje ga je spriječilo u ispunjavanju zahtjeva (slika 7.).



Slika 7. Prikaz greške nakon pogrešno unesenog upita

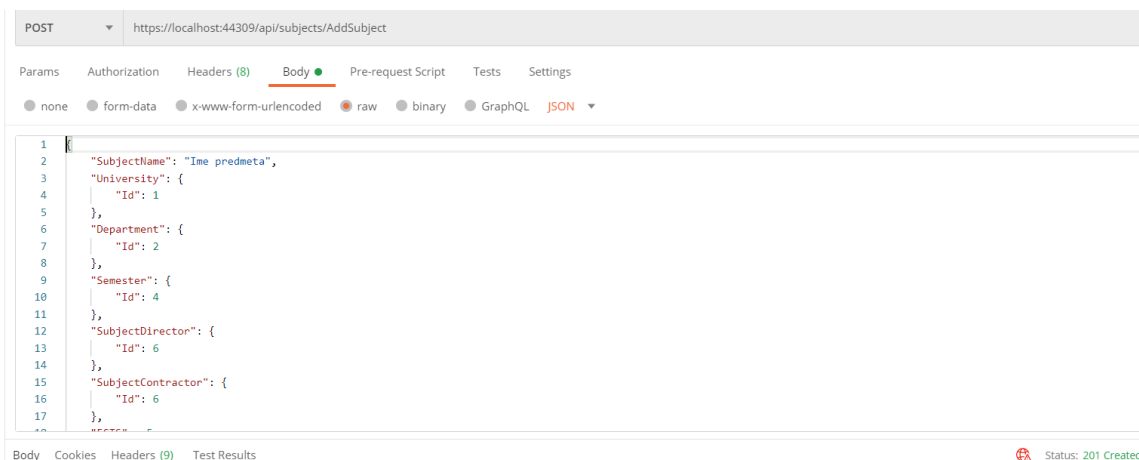
- **POST** - stvaranje zapisa

POST metoda se koristi za slanje podataka na poslužitelj za stvaranje ili ažuriranje resursa. Podaci poslani poslužitelju s POST metodom pohranjuju se u tijelu zahtjeva (engl. *request body*):

https://localhost:44309/api/subjects/AddSubject

URL prikazuje dodane podatke, u ovom slučaju dodaju se novi podaci za *AddSubject*. Podaci poslani POST metodom prolaze kroz HTTP zaglavlja tako da sigurnost ovisi o HTTP protokolu. Korištenjem sigurnosnog HTTP-a (HTTPS) možemo zaključiti da su podaci sigurni. POST metoda može se koristiti za američki standardni znakovnik za razmjenu informacija (engl. *American Standard Code for Information Interchange – ASCII*) kao i binarnih podataka. Na slici 8 vidljivi su parametri *Subjects*-a te dodane vrijednosti obveznim poljima u bazi (engl. *not null*) kao što su: *subject_name*, *university_id*, *department_id*, *semester_id*, *subject_director* te ostala polja u tablici SQLite baze zapisana u JSON obliku. Najčešći prikazani HTTP status kodovi su *404 (Not Found)* i *409 (Conflict)* ako

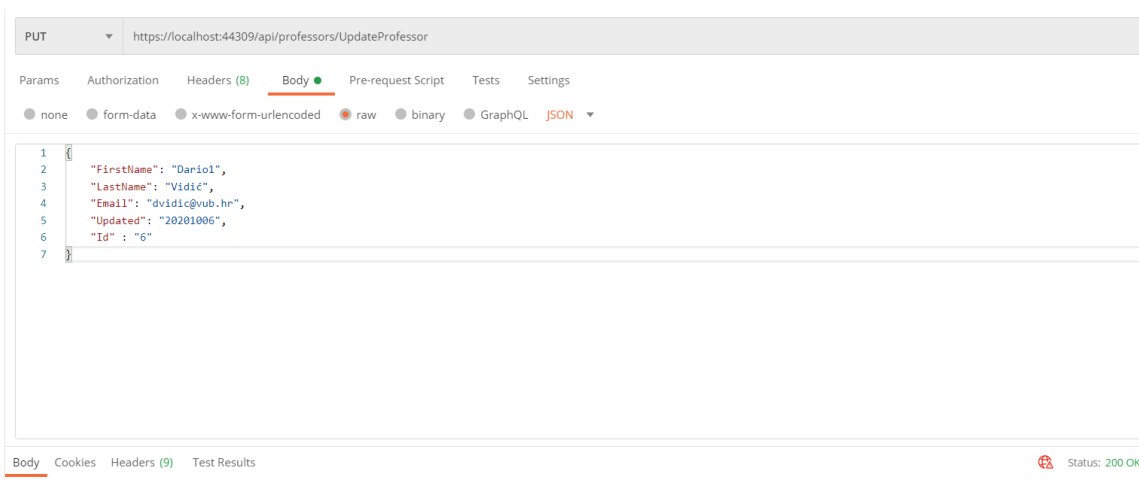
podaci već postoje, dok je u našem slučaju vidljivo na slici 8. HTTP status kod *201 Created* odnosno prikazuje dodane podatke.



Slika 8. Prikaz HTTP POST metode i parametara u JSON obliku

- **PUT** - promjene zapisa

PUT se najčešće koristi za ažuriranje postavki podataka, u našem primjeru za postavku promjene podataka profesora. Profesor može izmijeniti barem jedan podatak, primjerice email i ažurirati podatke koji se automatski ažuriraju i u SQLite bazi podataka. Najčešći HTTP status kod za PUT metodu su : *200 (engl. OK)* ili *204 (engl. No Content)*.

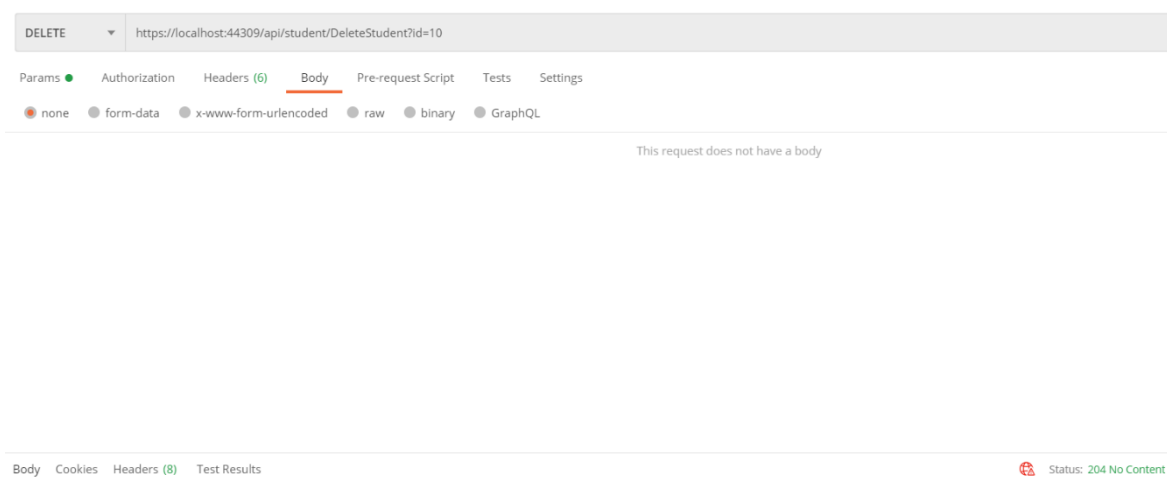


Slika 9. Prikaz HTTP PUT metode, JSON parametara i statusa

Unutar Postmana zapisujemo podatke u JSON formatu za POST i PUT metode. U našem primjeru na slici 9 prikazano je ažuriranje *FirstName* polja u *professor* tablici.

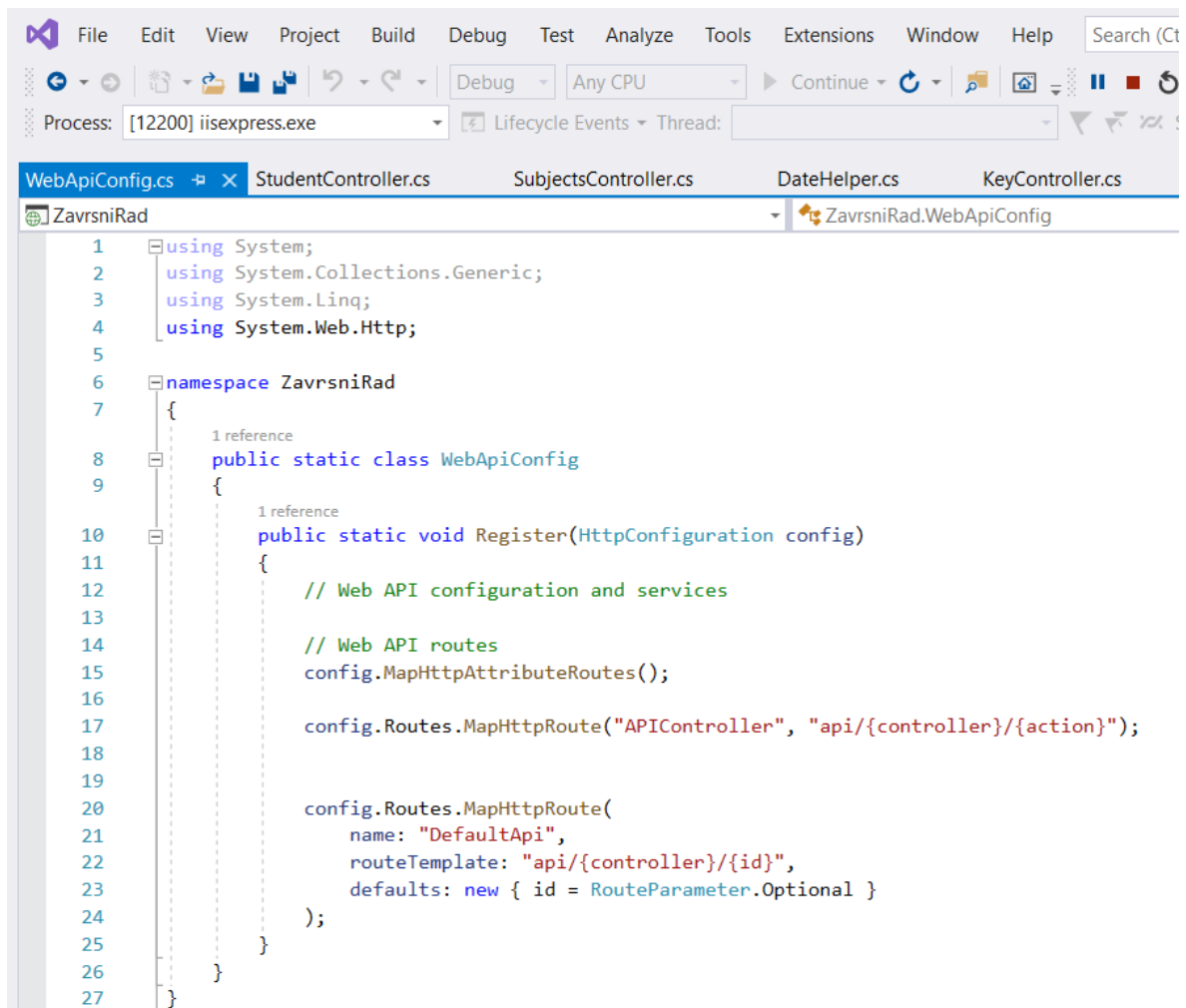
- **DELETE** - brisanje zapisa

Metoda DELETE HTTP koristi se za brisanje podataka na čiji URL korisnik šalje zahtjev. U slučaju uspješnog brisanja studenta s upitom *?id=10* vidljivo na slici 10. metoda DELETE HTTP vraća HTTP status kod *204 Nema sadržaja* (engl. *204 No Content*).



Slika 10. Prikaz HTTP DELETE metode i njezinog statusa

Detaljnije ćemo proučiti putanje (engl. *routes*). Inicijalna ruta prikazane web aplikacije jest: `api/{controller}/{action}` vidljiva na slici 11.



```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Web.Http;
5
6 namespace ZavrnsniRad
7 {
8     1 reference
9     public static class WebApiConfig
10    {
11        1 reference
12        public static void Register(HttpConfiguration config)
13        {
14            // Web API configuration and services
15
16            // Web API routes
17            config.MapHttpAttributeRoutes();
18
19            config.Routes.MapHttpRoute("APIController", "api/{controller}/{action}");
20
21            config.Routes.MapHttpRoute(
22                name: "DefaultApi",
23                routeTemplate: "api/{controller}/{id}",
24                defaults: new { id = RouteParameter.Optional }
25            );
26        }
27    }
```

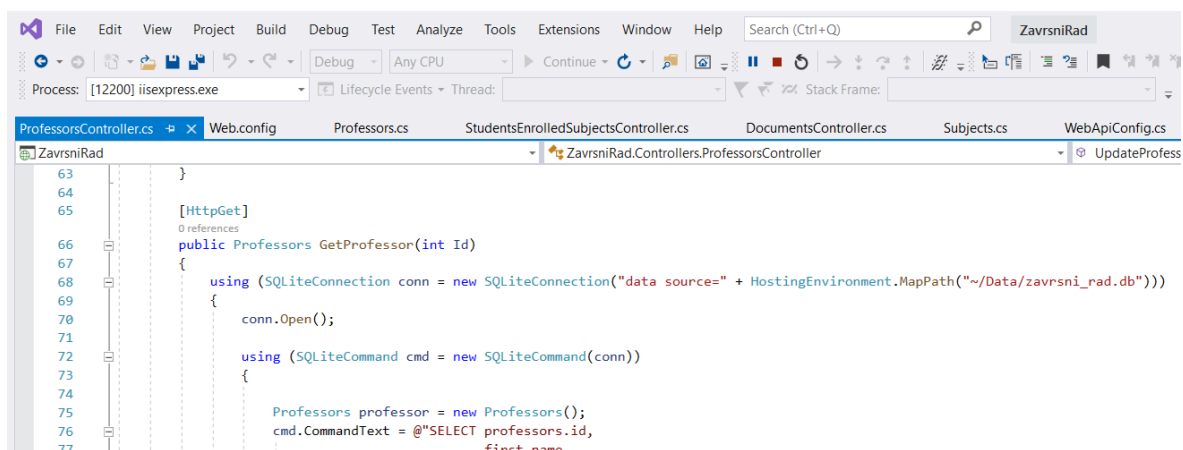
Slika 11. WebApiConfig.cs i inicijalnog URL-a

URL (engl. *Uniform Resource Locator*) ili jedinstveno mjesto izvora podataka je adresa određenog jedinstvenog resursa na Internetu. Pošto svaki valjani URL upućuje na jedinstveni resurs, korisnik može upisati URL i doći do krajnjeg željenog odredišta. Unutar WebApiConfig.cs-a (slika 11.) nalazi se URL vidljiv sa slike koji korisnik može unijeti u svoj preglednik kako bi pronašao aplikaciju. Prilikom kreiranja aplikacije generira se port te prilikom pokretanja aplikacije ona se izvodi s URL-om *http://localhost:44309*. Na idućem primjeru bit će pojašnjeni dijelovi URL-a:

https://localhost:44309/api/professors/GetProfessor?Id=2

- 1) *https* ili *http* → protokol
- 2) *localhost:44309/api* → URL adresa API-a
- 3) */professors/GetProfessor* → API metoda
- 4) *?Id=2* → upit

Prikazana putanja podijeljena je razdvojnim znakom (engl. *slash*) na više razina, točnije lokaciju podatka/resursa. Pomoću navedenog URL-a, korisniku će biti vidljivi podaci profesora s id-om 2. Upit se mijenja po želji korisnika te ovisno o kreiranim metodama i njihovim ulaznim parametrima. URL adresa prikazuje API unutar kojeg je kreiran kontroler (engl. *controller*) Professors koji sadrži metodu *GetProfessor* s ulaznim parametrom *Id* što je vidljivo iz slike 12. Ako korisnik ne napiše ispravan upit, web putanja je nepostojeća. Ako je *Id* profesora nula (engl. *null*) ili uneseni broj (engl. *integer*) ne postoji u SQLite bazi podataka javlja se poruka *Nisu pronađeni podaci* (engl. *no data found*).



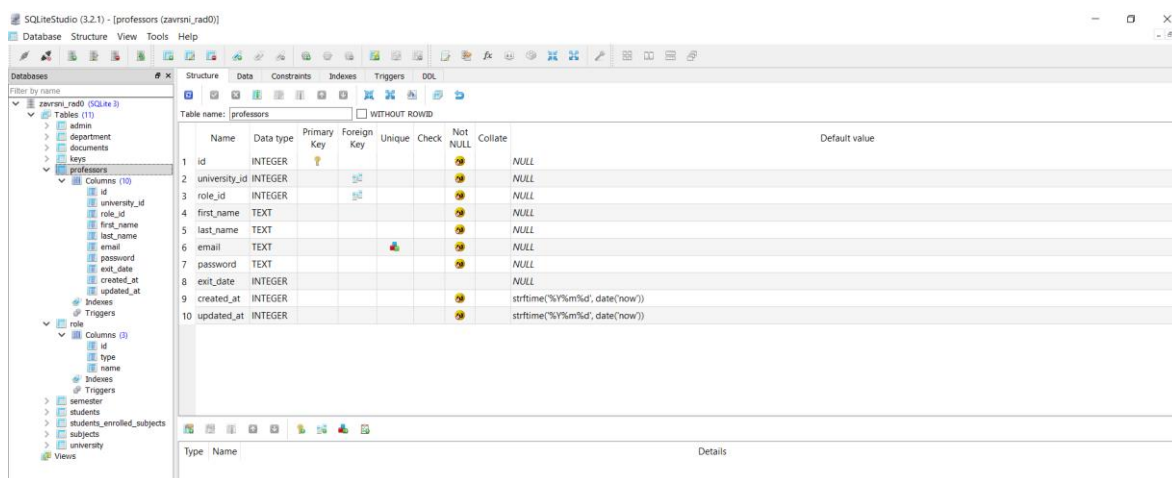
```
63     }
64
65     [HttpGet]
66     public Professors GetProfessor(int Id)
67     {
68         using (SQLiteConnection conn = new SQLiteConnection("data source=" + HostingEnvironment.MapPath("~/Data/zavrnsni_rad.db")))
69         {
70             conn.Open();
71
72             using (SQLiteCommand cmd = new SQLiteCommand(conn))
73             {
74
75                 Professors professor = new Professors();
76                 cmd.CommandText = @"SELECT professors.id,
77                                 first name,
```

Slika 12. Prikaz naziva kontrolera, metode i ulaznih parametara

6.2 Baza podataka

SQLite je relacijska baza podataka kreirana tako da bude samostalna te bez potreba za konfiguracijom. Cijela baza podataka smještena je u jednoj datoteci te je tako moguće dijeliti bazu između više računala. SQLite baza podataka je odlična za korištenje na web stranicama s malom ili srednjom količinom podataka. SQLite je dostupan na UNIX-u (Linux, Android, iOS, Mac OS-X) i Windowsima (Win32, WinCE, WinRT). Na slici 13. prikazana je SQLite baza koja je kreirana za potrebe aplikacije završnog rada. Jednostavna i funkcionalna baza s

jasnim pregledom tablica i strukturom kreiranih podataka sadrži sve što je potrebno za nesmetan rad. [11]

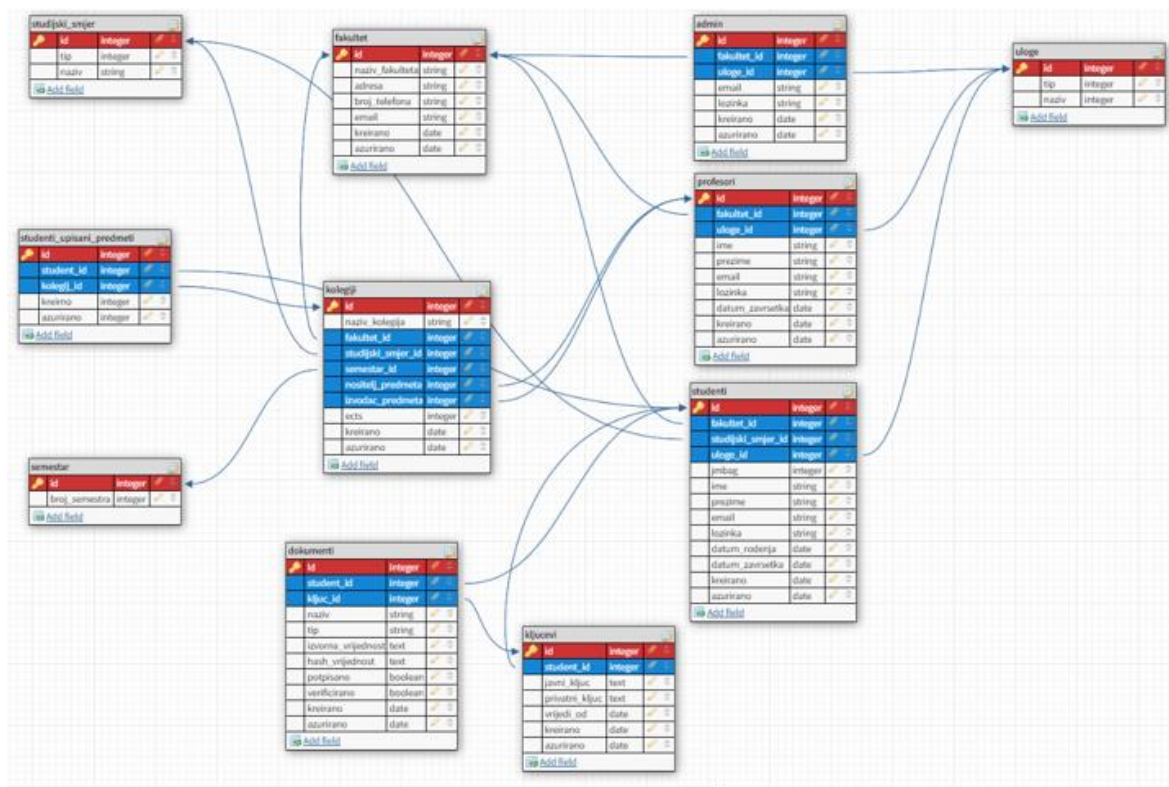


Slika 13. Prikaz SQLite baze podataka i radnog stabla aplikacije

6.2.1 Dizajn baze podataka

Za izradu modela baze podataka korištena je web aplikacija DbDesigner. Na slici 14 prikazan je model baze podataka sa svim entitetima i njihovim vezama.

Atributi čija je zadaća jedinstveno (engl. *unique*) opisati svaki od navedenih entiteta naziva se primarni ključ tablice (engl. *primary key*). Primarni ključ upotrebljava se za povezivanje tablice i ima dvostruku ulogu: jednoznačno definira redak tablice, a preko njega se ostvaruje i veza s drugim tablicama (na slici 14. polja crvene boje). [12] Uvrstimo li u drugu tablicu primarni ključ iz prve tablice moguće je referenciranje na tablicu koja je izvor primarnog ključa. U tim tablicama polja se nazivaju vanjskim ključem (na slici 14. polja plave boje). Vanjski ključ (engl. *foreign key*) je jednostavno rečeno, primarni ključ druge tablice.



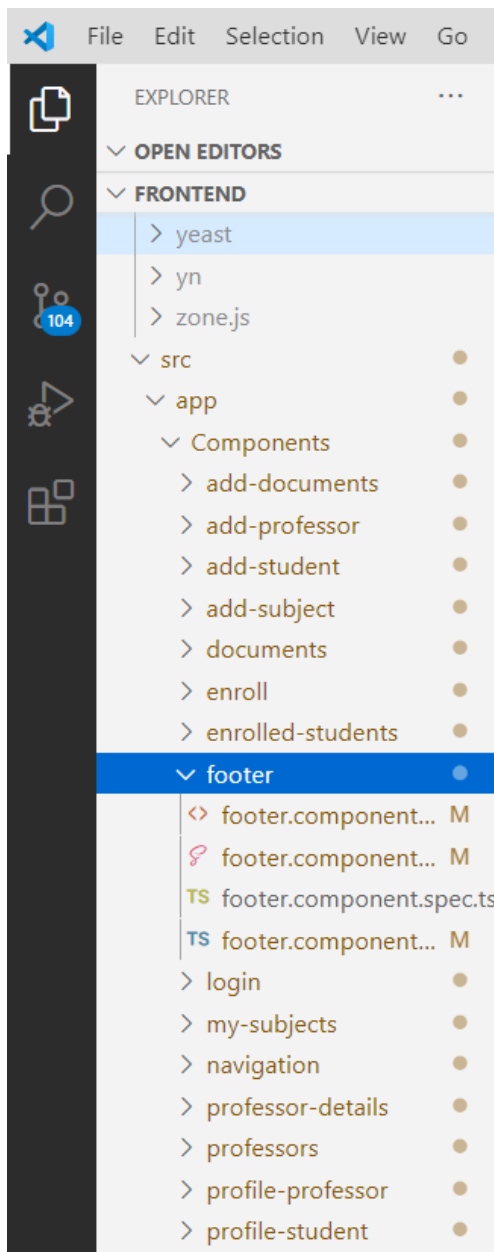
Slika 14. Prikaz svih tablica u bazi prikaze pomoću DbDesigner-a

Model (slika 14.) sastoji se od 11 međusobno povezanih tablica koje su povezane jedinstvenim primarnim i vanjskim ključem. Tako je moguće dohvatiti podatke iz drugih tablica koji odgovaraju fizičkoj implementaciji u SQLite bazi.

6.3 AngularJS

AngularJS je strukturalni okvir (engl. *framework*) za dinamičke web aplikacije. Angular je dobio naziv po izlomljenim zagradama kao važnom dijelu sintakse brojnih programskih jezika. Omogućuje uporabu HTML-a kao jezika predložka (engl. *template language*) te omogućuje proširivanje sintakse HTML-a na jasniji način. AngularJS razvijen je od strane Googlea 2010.godine zasnovan na JavaScript-u. AngularJS je potpuno besplatan otvoreni izvor (engl. *open source*) za kreiranje web aplikacija velikih razmjera, visokih performansi te jednostavan za održavanje. [13]

U radnom stablu (slika 15.) AngularJS aplikacije kreirane su tri mape unutar *add* datoteke: Components, Models i Services od strane korisnika aplikacije.



Slika 15. Prikaz radnog stabla unutar AngularJS-a s primjerom komponenta

Izrađena aplikacija može se podijeliti u nekoliko važnih strukturalnih cjelina:

1. **komponenta** (engl. *components*)

Svaku komponentu možemo zamisliti kao zasebnu cjelinu. Na slici 15. prikazane su komponente gdje na vrlo jasan i pregledan način možemo pronaći željenu komponentu. Komponenta se kreira u naredbenom retku (engl. *command prompt*) naredbenom linijom:

ng generate component ime

Unutar radnog stabla (engl. *working tree*) nakon dodavanja nove komponente ona poprima naziv u zelenoj boji te inicijalne .html, .scss, .ts i .spec.ts datoteke. Nazivi komponenti u AngularJS-u poredani su abecednim redoslijedom te tako dodatno olakšavaju preglednost.

2. **modeli** (engl. *models*)

U modelu se koriste objekti koji imaju sadržane podatke iz API-a. Preporučeno je da bi modeli trebali biti pohranjeni u zajedničkoj mapi ako će se koristiti u cijeloj aplikaciji. Tipove varijabli deklariraju se u modelu koji se zatim ponovno može koristiti u ostatku aplikacije. Na taj način primorano je pridržavati se određene specifikacije podataka. Kreirani modeli su zasebne .ts datoteke.

3. **servisi** (engl. *services*)

Servisi unutar AngularaJS-a sadrže metode u kojima su podaci dostupni cijelo vrijeme. Glavni zadatak servisa je organizirati modele i funkcije s različitim komponentama. Funkcije unutar servisa mogu pozvati iz bilo koje komponente te na taj način pomaže u dijeljenju web aplikacije na male i različite logičke jedinice koje se mogu iznova upotrijebiti.

```

src > app > Services > TS student.service.ts > StudentService
22  getStudent(id: number): Observable<Student> {
23      |   const url = `${this.StudentUrl}/GetStudent?Id=${id}`;
24      |   return this.http.get<Student>(url, this.httpOptions);
25      }
26
27  getStudents(): Observable<Student[]> {
28      |   const url = `${this.StudentUrl}/GetStudents`;
29      |   return this.http.get<Student[]>(url, this.httpOptions);
30      }
31
32  getStudentsByDepartment(departmentId: number): Observable<Student[]> {
33      |   const url = `${this.StudentUrl}/GetStudents?departmentId=${departmentId}`;
34      |   return this.http.get<Student[]>(url, this.httpOptions);
35      }
36
37  addStudent(student: Student): Observable<void> {
38      |   const url = `${this.StudentUrl}/AddStudent`;
39      |   return this.http.post<void>(url, student, this.httpOptions);
40      }
41
42  updateStudent(student: Student): Observable<Student>{
43      |   let url = `${this.StudentUrl}/UpdateStudent`;
44      |   return this.http.put<Student>(url, student, this.httpOptions);
45      }
46
47  deleteStudent(id: number): Observable<string> {
48      |   const url = `${this.StudentUrl}/DeleteStudent?Id=${id}`;
49      |   return this.http.delete<string>(url, this.httpOptions);
50      }
51  }

```

Slika 16. Prikaz metoda kreiranih u REST API-u te njihova implementacija unutar AngularJS-a

6.4 Zahtjevi na sustav

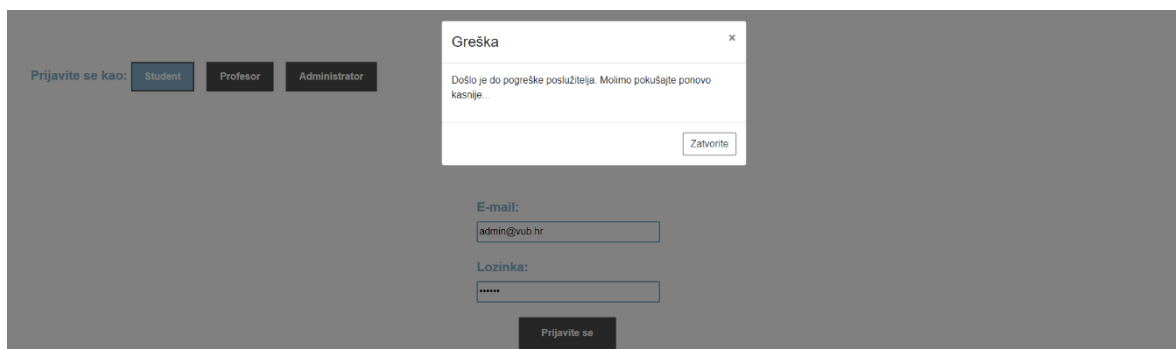
Funkcionalnost same aplikacije prikazana je određenim cjelinama i njihovim mogućnostima pri određenim pravima korisnika:

- Prijava u sustav
- Unos novih kolegija, profesora i studenata
- Pregled svih upisanih studenata i profesora
- Pregled vlastitih profila s mogućnošću izmjene podataka
- Predaja kolokvija i njihovo preuzimanje
- Automatizirano potpisivanje dokumenata te njihovo verificiranje

Prilikom prijave u sustav student prilaže kolokvij u predviđenu rubriku te se automatski generira privatni ključ koji garantira autentičnost kolokvija. Profesor se prijavljuje u aplikaciju s višom razinom prava u odnosu na studenta te ima mogućnost preuzeti kolokvij i verificirati ga.

7. FUNKCIONALNOST APLIKACIJE I KORISNIČKO SUČELJE

Prilikom prijave u aplikaciju korisnik se susreće s formom prijave u sustav ovisno o razini prava. Prijava je moguća pomoću e-maila i lozinke nakon što se odabere tip (student, profesor i administrator) prijave. Ovisno o tipu prijave prikazani su različiti prikazi aplikacije, ali i različita prava nad samom aplikacijom. U slučaju pogrešno unesenih podataka prikazuje se skočni prozor (engl. *popup*) s odgovarajućom porukom korisniku. Skočni prozor pomaže prijavljenom korisniku ako korisnik primjerice unese krive podatke. Pojavljuju se bez korisnikova dopuštenja i pomažu u očuvanju funkcionalnost aplikacije. Svrha skočnog prozora je obavijestiti korisnika o određenim promjenama u aplikaciji kako bi bili svjesni kako njihov odabir može utjecati na aplikaciju. U primjeru vidljivom na slici X prikazana je greška prilikom prijave u aplikaciju. Korisnik prilikom prijave nije unesao ispravne podatke, te skočni prozor obavještava korisnika da je došlo do pogreške na poslužitelju. To znači da korisnik prilikom prijave nije pritisnuo ispravan oblik prijave kao administrator već je unesao podatke za prijavu kao da se prijavljuje student.



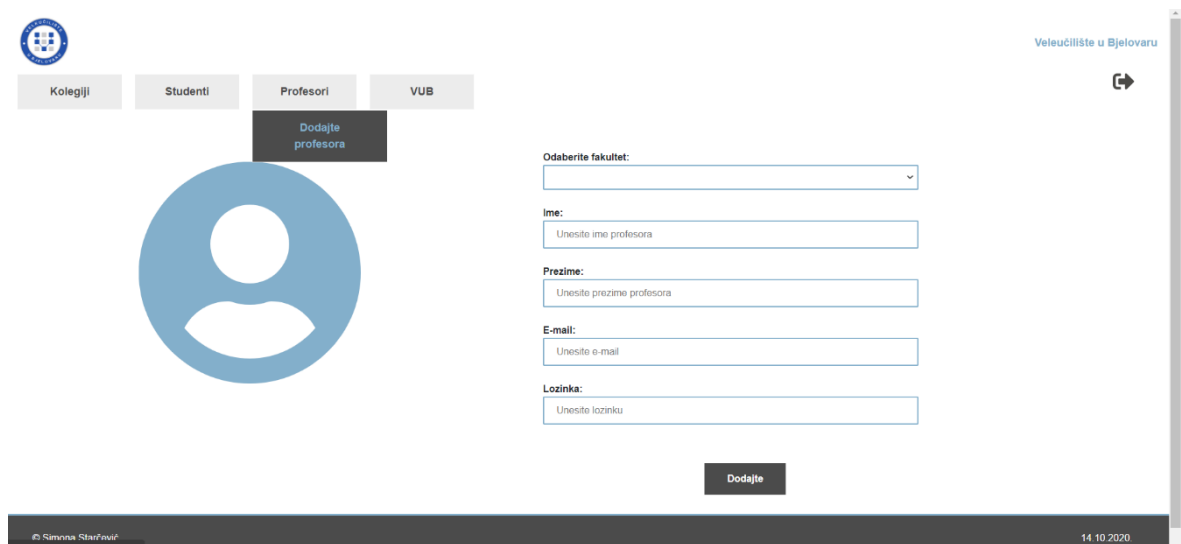
Slika 17. Prikaz skočnog prozora

Nakon uspješne prijave u aplikaciju pojavljuje se početno sučelje u skladu s razinom korisnikova prava te podnožje (engl. *footer*) koje prikazuje ime i prezime prijavljene osobe te današnje vrijeme (engl. *date now*). Nakon što je korisnik prijavljen u aplikaciju u gornjem desnom dijelu ekrana nalazi se ikona za odjavu (engl. *logout*) iz aplikacije.

7.1 Administrator

Funkcionalnost aplikacije vidljiva je na navigacijskoj traci koja određuje određene komponente kao cjeline na jednostavan i pregledan način. Na slici 17. prikazan je izgled navigacijske trake koju vidi osoba prijavljena s administratorskim pravima. Kao korisnik kojemu je povjeren pristup ograničenim alatima i mogućnostima dozvoljene su radnje kao dodavanje ili brisanje. Pod nazivom kartice (engl. *tab*) *Kolegiji* prikazan je padajući izbornik među kojem se nalazi mogućnost *Dodaj kolegij*. Ovdje se nalaze mogućnosti poput odabira visokog učilišta, stručnog studija, semestra i brojne druge. Ako korisnik nije siguran u točnost podataka, pritiskom na naziv unesenog predmeta vidi sve upisane podatke. Jedino administrator ima mogućnost brisanja i dodavanja predmeta. Brisanje se vrši pritiskom na ikonu kante za otpatke (engl. *recycle bin*).

Dodavanje studenta i profesora omogućeno je jedino administratoru. Ako administrator želi dodati profesora, na navigacijskoj traci odabire *Profesori* → *Dodaj profesora*, dok za dodavanje studenta odabire *Student* → *Dodaj studenta*. Kako bi se kreirao novi student ili profesor, administrator popunjava formu za unos podataka. Gumb (engl. *button*) *Dodajte* onemogućen (engl. *disable*) je sve dok se ne popune sva polja u formi.

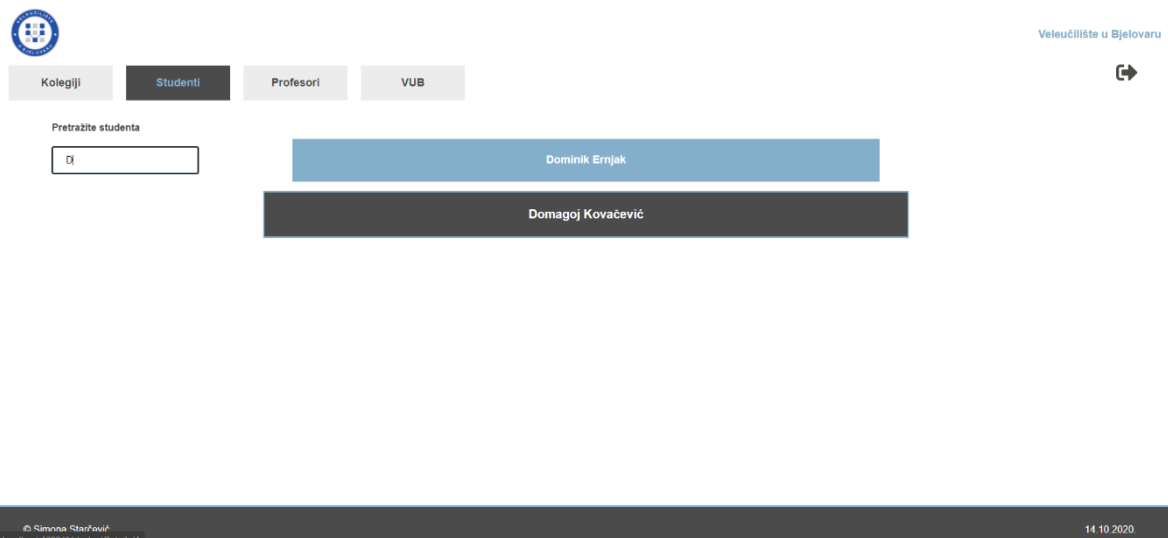


The screenshot shows the administrator interface for adding a professor. At the top, there is a navigation bar with four tabs: 'Kolegiji', 'Studenti', 'Profesori', and 'VUB'. The 'Profesori' tab is selected, and a dark button labeled 'Dodajte profesora' is positioned over it. Below the navigation bar is a large blue circular icon with a white silhouette of a person's head and shoulders. To the right of the icon is a form with the following fields: 'Odaberite fakultet:' (a dropdown menu), 'Ime:' (a text input field with the placeholder 'Unesite ime profesora'), 'Prezime:' (a text input field with the placeholder 'Unesite prezime profesora'), 'E-mail:' (a text input field with the placeholder 'Unesite e-mail'), and 'Lozinka:' (a text input field with the placeholder 'Unesite lozinku'). At the bottom center of the form is a dark button labeled 'Dodajte'. The footer of the page contains the text '© Simuna Starčević' on the left and '14.10.2020' on the right.

Slika 18. Administrator programsko sučelje

Kada administrator odabere *Studenti* ili *Profesori* na navigacijskoj traci, prikazuje se popis svih studenata, odnosno profesora. Prikazan je abecedni popis svih studenata uzlazno po prezimenu (engl. *ascending* - *ASC*). Vidljiva je tražilica koja prilikom unosa teksta filtrira

osobu po imenu i/ili prezimenu te je tako pronalazi (slika 18.). Funkcionalnost pretrage moguća je nad studentima, profesorima i predmetima radi brže pretrage istih.



Slika 19. Administrator - pretraga studenata

Administrator nema mogućnost uvida u profil profesora i studenta te je iz tog razloga realizirana mogućnost pritiska na ime i prezime korisnika kako bi bili prikazani njegovi detalji. Ako postoje dva studenta s istim imenom i prezimenom, bit će prikazana oba. Jedinstveni podaci za unos studenta su JMBG i e-mail, dok je za profesora e-mail. Unesu li se dva korisnika s istim podacima prikazat će se skočni prozor s upozorenjem te u konačnici nemogućnost dodavanja. Na navedenom primjeru zadovoljena je autentičnost korisnika što je od iznimne važnosti za daljnji rad aplikacije.

Navigacijska traka svih korisnika sadrži kategoriju VUB koja sadrži osnovne informacije kao što su telefonski brojevi i e-mail adrese važnih fakultetskih organizacija. Korisne informacije koje su od bitne važnosti uvijek su na raspolaganju svim korisnicima.

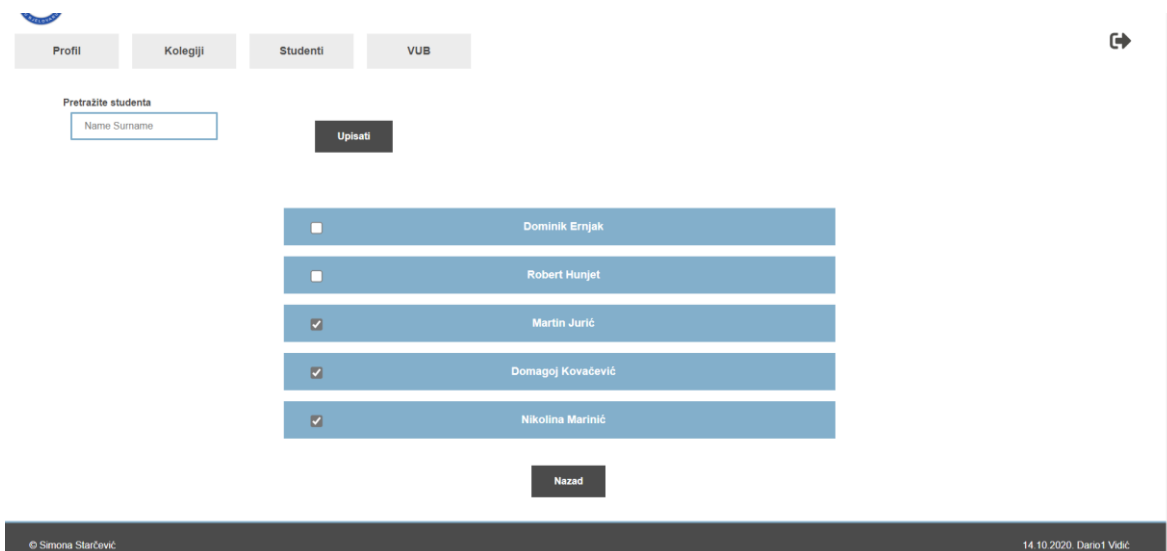
7.2 Profesor

Za razliku od administratora, profesor ima manju razina prava koja sadrži sve informacije bitne za profesora. Profesoru je vidljiv vlastiti profil odmah prilikom prijave u aplikaciju te osobne podatke može mijenjati i spremi promjene. Ako korisnik pritisne na *Izmijeni* “otključava” se forma s podacima i moguće je promijeniti osobne podatke. Kada su

podaci promijenjeni gumb mijenja naziv u *Spremi* te se izmijenjeni podaci spremaju u bazu. Prikazuje se skočni prozor s porukom da je unos novih podataka uspješno odrađen.

Pritiskom na *Kolegiji* → *Moji kolegiji* prikazana je lista predmeta osobno za svakog profesora te klikom na određeni predmet vidljiv je prikaz upisanih studenata na određeni predmet. Prikazana je lista upisanih studenata pored kojih se nalazi ikona kante za otpatke na čiji se pritisak briše određeni student. Pritiskom na studenta koji se nalazi na popisu upisanih studenata nalaze se njegovi osobni podaci (JMBAG, e-mail adresa, studijski smjer itd.), ali i priloženi kolokviji koje profesor verificira. Kolokviji su prilikom pisanja potpisani od strane studenta njegovim privatnim ključem dok ga profesor može čitati uporabom javnog ključa i tako svjedočiti autentičnost kolokvija/ispita.

Funkcionalnost po kojoj je profesor prepoznatljiv svakako je dodavanje, odnosno upisivanje studenata na vlastiti kolegij te verifikacija studentovih kolokvija. Moguća je pojava dvaju ili više istih naziva predmeta, ali se razlikuju po studijskom smjeru koji je detaljno naznačen kako bi se olakšalo snalaženje. Klikom na *Upiši studenta* prikazuje se sučelje prikazano na slici 17. gdje je dodan popis trenutno neupisanih studenata s potvrdnim okvirom (engl. *checkbox*) kako bi se olakšao višestruki odabir studenata za upis na određeni kolegij. Nakon odabranih studenata za upis na kolegij, profesor potvrđuje njihov upis klikom na *Upisati*, a želi li odustati potrebno je kliknuti na gumb *Nazad*. Upisani studenti sada se nalaze na popisu upisanih studenata željenog kolegija i ne prikazuju se listi.

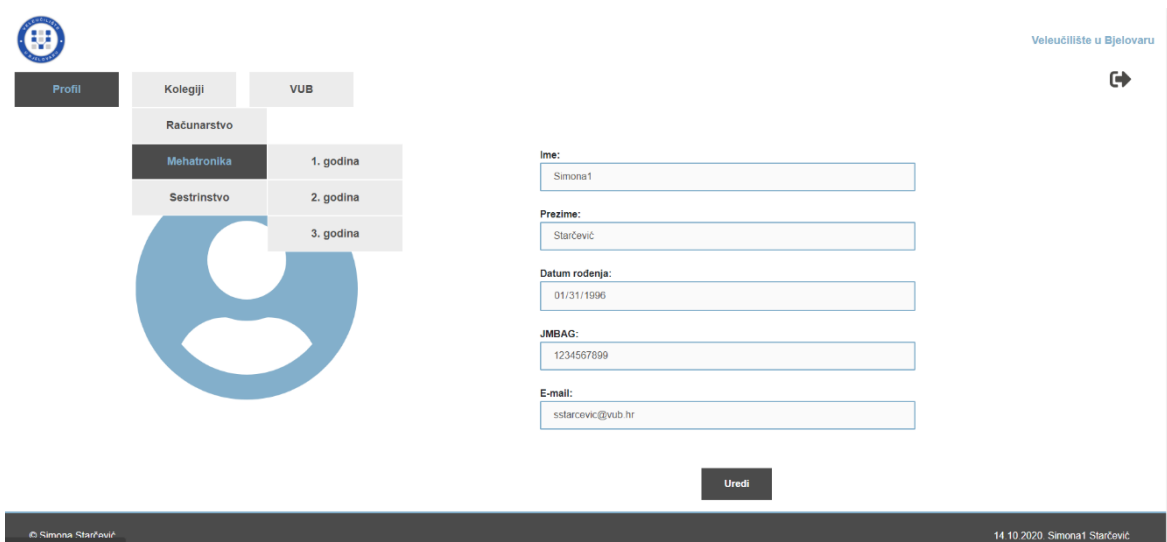


Slika 20. Profesor - odabir studenata za upis u kolegij

7.3 Student

Student je korisnik aplikacije koji ima najmanju razinu prava radi prikazivanja neželjenih informacija zbog sigurnosti i povjerljivosti podataka. Ulogirani student dolazi na početnu stranicu gdje su prikazani njegovi osobni podaci te mu je ponuđena mogućnost izmjene istih. Pritiskom na gumb (engl. *button*) „Uredi“ omogućena je izmjena osobnih podataka studenta te izmjena. Nakon što je student izmijenio podatke i pritiskom na gumb „Promijeni“ prikazuje se skočni prozor s porukom o uspješno izmijenjenim podacima. Lijevi dio ekrana predviđen je za fotografiju korisnika, što se u budućoj nadogradnji projekta može realizirati povlačenjem podataka iz ISVU sustava (studentska iskaznica).

Kategorija *Predmeti* smještena na navigacijskoj traci korisničkog sučelja prikazuje popis svih predmeta u ovisnosti o studijskom smjeru i godini. Svaka studijska godina podijeljena je u dva stupca, odnosno na zimski i ljetni semestar zajedno sa popisom kolegija. Pritiskom na određeni predmet prikazuju se detalji tog predmeta s najvažnijim informacijama za samog studenta (slika 20).



Slika 21. Student - prikaz profil studenta s osobnim podacima i navigacijske trake

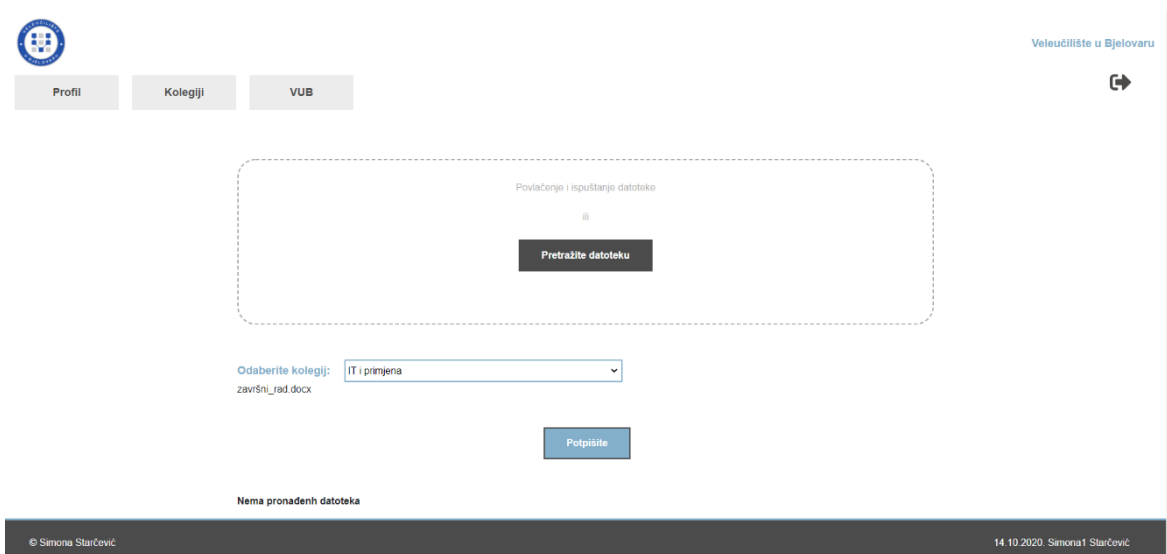
Pod *Profil* → *Dokumenti* nalazi se najvažnije sučelje za studenta u kojemu je moguće priložiti napisani kolokvij. Pomoću padajućeg izbornika (engl. *drop down*) student odabire kolegij za koji želi predati dokumente. Student prilaže dokument pomoću dva načina:

- povucite i ispustite (engl. *drag and drop*) - jednostavan način na koji korisnik odabire datoteku te je pritisne i prevuče na željenu lokaciju (slika 21.)
- klikom na gumb Pretraži (engl. *browse*) - pritiskom na gumb *Pretraži* otvara se istraživač datoteka (engl. *file explorer*) te nakon što korisnik pronade željeni dokument pritisne na gumb *Otvori* (engl. *open*)

Moguće je dodati različite vrste dokumenata poput .pdf, .png, .jpg, docx i mnoge druge. Student nakon što je odabrao željeni dokument odabire kolegij pomoću padajućeg izbornika. Prije nego *Potpíše* dokument/e ima mogućnost dodavanja ili uklanjanja dokumenata koji čekaju potpisivanje. Kada se dokument potpiše nalazi se na listi potpisanih dokumenata te tako student i profesor u bilo kojem trenutku mogu pristupiti dokumentima. U listi potpisanih dokumenata nalaze se ikonice *Preuzimanje* (engl. *download*) u oblike strelice koja pokazuje prema dolje i *Verificiraj* (engl. *verify*), ikona u obliku kvačice.

Preko odabranog predmeta dolazi do “komunikacije” između studenta i profesora. Drugim riječima, student može izabrati samo one predmete koje je upisao dok profesor može vidjeti dokumente studenata kojima predaje.

Radi lakšeg razumijevanja aplikacije na primjeru će se prikazati interakcija između studenata i profesora na primjeru pisanja kolokvija. Kada student napiše kolokvij dužan ga je uslikati ili skenirati. Nakon prijave u aplikaciju prilaže kolokvij unutar okvira predviđenog za predaju kolokvija. Odabire kolegij u padajućem izborniku te student može odabrati jedino upisane kolegije u tekućoj akademskoj godini. Prikazuje se naziv dokumenta i ekstenzija ovisno o nazivu i tipu dokumenta. Ako je student siguran u odabir dokumenta potpisuje (engl. *sign*) kolokvij. Zahvaljujući prijavi u sustav omogućena je autorizacija studenta te autentičnost dokumenta. Kako bi profesor potvrdio autentičnost dokumenata prijavljuje se u sustav i pod vlastitim predmetima pronalazi predane dokumente. Odabirom dokumenta koji želi provjeriti otvara mu se stranica s detaljima o digitalnom potpisu dokumenta gdje su prikazani ime i prezime potpisnika i vrijeme kada je dokument potpisan (engl. *timestamp*).



Slika 22. Student - predaja kolokvija i njegovo potpisivanje

8. ZAKLJUČAK

Najvažniji segment izrađene aplikacije je održavanje sigurnosti podataka što je uspješno odrađeno prateći smjernice i literaturu kolegija Sigurnost i zaštita podataka. Trend digitalnog obrazovanja u konstantnom je porastu i smatra se dugoročnim oblikom edukacije. Osim predaje i verificiranih pisanih provjera znanja, moguće je aplikaciju u daljnjem razvoju nadograditi i za pisanje provjera znanja. U tom slučaju bi aplikacija bila na razini trenutnih sustava za upravljanje učenjem. Vrijeme prilagodbe digitalnog načina obrazovanja u vrijeme pisanja završnog rada bilo je nametnuto i nepredvidljivo. Aplikacija je kreirana na jasan i razumljiv način kako bi njeni korisnici svu pažnju usmjerili najvažnijem cilju - savladavanju gradiva. U trenutnoj epidemiološkoj situaciji javlja se potreba za zaštitom podataka te mogući nastanak boljih i bržih algoritama, ali i sigurnosnih programa protiv krađe podataka i neovlaštenih pristupa. Duljine ključeva postojat će sve veće i sve teže za dešifrirati kako bi se što više aktivnosti odvijalo na daljinu bez straha od sigurnosti mreže i samih podataka. Poboljšanje kreirane aplikacije je internacionalizacija jezika kako bi aplikacija bila uporabljiva i prilagodljiva izvan Hrvatske. U aplikaciji nisu prikazane slike profesora i studenata čiji je prikaz u budućnosti planiran realizirati preko informacijskog sustava visokih učilišta (ISVU) sa studentskih iskaznica. Moguće je i prilagoditi aplikaciju korištenjem drugih asimetričnih algoritmima za potpisivanje. Isto tako, moguće je digitalno potpisivati kolokvije studenata korištenjem certifikata. Nedvojbene prednosti digitalnih platformi te informacijsko-komunikacijske tehnologije u cjelini pokazale su se tokom 2020. godine više nego ikad. No, osim tih prednosti koje su omogućile nastavak mnogih djelatnosti i napose obrazovanja, iznjedrili su se i neki nedostaci. Upravo jedan od tih je i predmet ovog završnog rada kojim bi se povećala sigurnost prilikom pisanja ispita u školama ili visokim učilištima. Sigurnost i vjerodostojnost su veliki izazovi, napose u digitalnom okruženju. Izrađena aplikacija u ovom završnom radu pokriva upravo taj aspekt, a njenim daljnjim razvojem za što postoji i veliki motiv, moglo bi se postići cjelokupno rješenje za provedbu pisanih ispita.

9. LITERATURA

[1] Softveri za provjeru autentičnosti radova [Online]. 2020.

Dostupno na: <https://www.srce.unizg.hr/spa>

[2] Provjera i vrednovanje ishoda učenja u online okruženju – savjeti i preporuke [Online].

Dostupno na:

https://www.inf.uniri.hr/images/naslovnica/2020/Preporuke_vrednovanja_ishoda_ucenja_u_online_okruzenju.pdf

[3] Digitalno obrazovanje: rizici online učenja [Online]. 2020.

Dostupno na: <https://www.cert.hr/digitalno-obrazovanje-kiberneticki-rizici-online-ucenja/>

[4] Klasična kriptografija [Online].

Dostupno na: <https://web.math.pmf.unizg.hr/~duje/kript/osnovni.html>

[5] Vidić D. Sigurnost računala i podataka: Uvod u kriptografiju, pdf. 2020.

[6] Korunić D. Sigurnost elektroničke pošte i PGP [Online]. 2002.

Dostupno na: <https://dkorunic.net/pdf/Sigurnost-poste-PGP.pdf>

[7] Symmetric cryptography [Online].

Dostupno na:

https://www.ibm.com/support/knowledgecenter/SSB23S_1.1.0.2019/gtps7/s7symm.html

[8] J. Ahmeti Kriptografija, pdf. 2011.

[9] Digital Signatures, [Online]. 2019.

Dostupno na: <https://cryptobook.nakov.com/digital-signatures>

[10] REST API, [Online].

Dostupno na:

https://www.ibm.com/support/knowledgecenter/hr/SSLKT6_7.6.1/com.ibm.mif.doc/gp_int_frmwk/rest_api/c_rest_overview.html

[11] SQLite, službena stranica [Online]. 2020.

Dostupno na: <https://www.sqlite.org/limits.html>

[12] Baze podataka, [Online].

Dostupno na: https://e-u.hr/dok/udzbenik/31_210.pdf

[13] 5 Awesome AngularJS Features, [Online]. 2012.

Dostupno na: <https://code.tutsplus.com/tutorials/5-awesome-angularjs-features--net-25651>

10. OZNAKE I KRATICE

AES - engl. *Advanced Encryption Standard*
API - engl. *application programming interface*
ASC – engl. *ascending*
DES - engl. *Data Encryption Standard*
DDoS - engl. *Distributed Denial-Of-Service*
DS – engl. *digital signature*
DSA – engl. *Digital Signature Algorithm*
ECC – engl. *Elliptic-Curve Cryptography*
ECDSA – engl. *Elliptic Curve Digital Signature Algorithm*
engl. – engleski
HTML - engl. *HyperText Markup Language*
HTTP – engl. *HyperText Transfer Protocol*
ISVU - informacijski sustav visokih učilišta
itd. – i tako dalje
JMBG - jedinstveni matični broj građana
JSON - engl. *JavaScript Object Notation*
LMS - engl. *Learning Management System*
MD5 - engl. *Message Digest Algorithm 5*
npr. – na primjer
PGP - engl. *Pretty Good Privacy*
REST - engl. *Representational State Transfer*
RSA - Rivest-Shamir-Adleman
SHA-1, SHA-2, SHA-3 - engl. *Secure Hash Algorithm*
SSL - engl. *Secure Sockets Layer*
TLS - engl. *Transport Layer Security*
UNIX - engl. *Uniplexed Information and Computing System*
URL - engl. *Uniform Resource Locator*
3DES - engl. *Triple Data Encryption Standard*

11. SAŽETAK

Naslov: Tehnike digitalnog potpisa u primjeni potpisivanja pisanih provjera znanja

Korištenjem SQLite baze podataka, Angular-a za Web aplikaciju, te .NET Framework (C#) REST API-a kreiran je sustav za generiranje i dohvaćanje digitalnog potpisa, te potpisivanje kolokvija. Objedinjeni su osnovni segmenti digitalnog potpisa: autentičnost, autorizacija i verifikacija. Cilj rada je omogućiti studentima kreiranje ključa za potpisivanje kolokvija i potvrđivanje identiteta studenta. Studentima se tako otežava mogućnost kršenja akademske čestitosti tijekom kolokvija u uvjetima rada i studiranja od doma, a profesorima je omogućena provjera identiteta studenata kroz digitalni potpis, te verifikacija valjanosti potpisanih kolokvija.

Ključne riječi: učenje na daljinu, digitalni potpis, REST API, AngularJS.

12. ABSTRACT

Title:

Digital signature techniques in the application of signing written knowledge tests

Using SQLite database, Angular for web application, .NET Framework (C #) REST API, a system was created for generating and retrieving digital signature, exam signing. The basic segments of digital signature are united: authenticity, authorization and verification. Goal of the work is to enable students to create a key for signing the colloquium and confirming the student's identity. Students are prevented from copying and assisting a third party during the colloquium in working and studying conditions from home, and professors are enabled to verify the identity of students through a digital signature, and to verify the validity of signed exam.

Keywords: distance learning, digital signature, REST API, AngularJS.

13. PRILOZI

Popis tablica

Slika 1. Model simetrične kriptografije	5
Slika 2. Model asimetrične kriptografije.....	6
Slika 3. Prikaz tijeka digitalnog potpisivanja	9
Slika 4. Prikaz vrijednosti MD5 hash algoritma.....	10
Slika 5. Dobivena vrijednost SHA-1 algoritma.....	10
Slika 6. Prikaz HTTP GET metode pomoću Postman-a	14
Slika 7. Prikaz greške nakon pogrešno unesenog upita	15
Slika 8. Prikaz HTTP POST metode i parametara u JSON obliku.....	16
Slika 9. Prikaz HTTP PUT metode, JSON parametara i statusa	16
Slika 10. Prikaz HTTP DELETE metode i njezinog statusa	17
Slika 11. WebApiConfig.cs i inicijalnog URL-a.....	18
Slika 12. Prikaz naziva kontrolera, metode i ulaznih parametara	19
Slika 13. Prikaz SQLite baze podataka i radnog stabla aplikacije	20
Slika 14. Prikaz svih tablica u bazi prikaze pomoću DbDesigner-a.....	21
Slika 15. Prikaz radnog stabla unutar AngularJS-a s primjerom komponenta.....	22
Slika 16. Prikaz metoda kreiranih u REST API-u te njihova implementacija unutar AngularJS-a	24
Slika 17. Prikaz skočnog prozora	25
Slika 18. Administrator programsko sučelje	26
Slika 19. Administrator - pretraga studenata.....	27
Slika 20. Profesor - odabir studenata za upis u kolegij.....	29
Slika 21. Student - prikaz profil studenta s osobnim podacima i navigacijske trake	30
Slika 22. Student - predaja kolokvija i njegovo potpisivanje.....	31

Popis tablica

Tablica 1. Performanse RSA i ECDSA algoritma.....	12
---	----

IZJAVA O AUTORSTVU ZAVRŠNOG RADA

Pod punom odgovornošću izjavljujem da sam ovaj rad izradio/la samostalno, poštujući načela akademske čestitosti, pravila struke te pravila i norme standardnog hrvatskog jezika. Rad je moje autorsko djelo i svi su preuzeti citati i parafraze u njemu primjereno označeni.

Mjesto i datum	Ime i prezime studenta/ice	Potpis studenta/ice
U Bjelovaru, <u>22.10.2020</u>	SIMONA STARČEVIĆ	Simona Starčević

Prema Odluci Veleučilišta u Bjelovaru, a u skladu sa Zakonom o znanstvenoj djelatnosti i visokom obrazovanju, elektroničke inačice završnih radova studenata Veleučilišta u Bjelovaru bit će pohranjene i javno dostupne u internetskoj bazi Nacionalne i sveučilišne knjižnice u Zagrebu. Ukoliko ste suglasni da tekst Vašeg završnog rada u cijelosti bude javno objavljen, molimo Vas da to potvrdite potpisom.

Suglasnost za objavljivanje elektroničke inačice završnog rada u javno dostupnom nacionalnom repozitoriju

SIMONA STARČEVIĆ

ime i prezime studenta/ice

Dajem suglasnost da se radi promicanja otvorenog i slobodnog pristupa znanju i informacijama cjeloviti tekst mojeg završnog rada pohrani u repozitorij Nacionalne i sveučilišne knjižnice u Zagrebu i time učini javno dostupnim.

Svojim potpisom potvrđujem istovjetnost tiskane i elektroničke inačice završnog rada.

U Bjelovaru, 22.10.2020

Simona Starčević
potpis studenta/ice