ŠOLSKI CENTER CELJE

Srednja šola za kemijo, elektrotehniko in računalništvo

E-dentiteta

Raziskovalna naloga

Avtorji:

Žan Škorja, R-3. b

Dominik Brezovšek, R-3. b

Nik Mejak, R-3. b

Mentor:

Boštjan Lubej, dipl. inž. inf. in tehn. kom.

Mestna občina Celje, Mladi za Celje

Celje, 2023

ŠOLSKI CENTER CELJE

Srednja šola za kemijo, elektrotehniko in računalništvo

E-dentiteta

Raziskovalna naloga

Avtorji:

Žan Škorja, R-3. b

Dominik Brezovšek, R-3. b

Nik Mejak, R-3. b

Mentor:

Boštjan Lubej, dipl. inž. inf. in tehn. kom.

Mestna občina Celje, Mladi za Celje

Celje, 2023

IZJAVA

Mentor Boštjan Lubej v skladu z 20. členom Pravilnika o organizaciji mladinske raziskovalne dejavnosti »Mladi za Celje« Mestne občine Celje, zagotavljam, da je v raziskovalni nalogi z naslovom E-dentiteta, katere avtorji so Žan Škorja, Dominik Brezovšek in Nik Mejak:

– besedilo v tiskani in elektronski obliki istovetno,

– pri raziskovanju uporabljeno gradivo navedeno v seznamu uporabljene literature,

– da je za objavo fotografij v nalogi pridobljeno avtorjevo dovoljenje in je hranjeno v šolskem arhivu,

– da sme Osrednja knjižnica Celje objaviti raziskovalno nalogo v polnem besedilu na knjižničnih portalih z navedbo, da je raziskovalna naloga nastala v okviru projekta Mladi za Celje,

– da je raziskovalno nalogo dovoljeno uporabiti za izobraževalne in raziskovalne namene s povzemanjem misli, idej, konceptov oziroma besedil iz naloge ob upoštevanju avtorstva in korektnem citiranju,

– da smo seznanjeni z razpisni pogoji projekta Mladi za Celje.

Celje, 13. 4. 2023

žig šole

ŠOLSKI CENTER CELJE
SREDNJA ŠOLA ZA KEMIJO, ELEKTROTEHNIKO IN RAČUNALNIŠTVO
CELJE, Pot na Lavo 22

Podpis mentorja

Podpis odgovorne osebe

POJASNILO

V skladu z 20. členom Pravilnika raziskovalne dejavnosti »Mladi za Celje« Mestne občine Celje je potrebno podpisano izjavo mentorja (-ice) in odgovorne osebe šole vključiti v izvod za knjižnico, dovoljenje za objavo avtorja (-ice) fotografskega gradiva, katerega ni avtor (-ica) raziskovalne naloge, pa hrani šola v svojem arhivu.

Zahvala

Zahvaljujemo se vsem, ki ste kakorkoli pomagali pri izdelavi raziskovalne naloge. Brez vaše pomoči naloga ne bi nastala, pa naj je šlo le za spodbudne besede, majhno idejo, nasvete ali kritike pri izdelovanju izdelka.

Najprej bi se zahvalili našemu mentorju profesorju Boštjanu Lubeju za ves trud, čas, podporo in vztrajnost, ki jo je vložil v izdelovanje raziskovalne naloge. Zahvalili bi se mu tudi za tehnični pregled naloge.

Profesorici Tjaši Verdev bi se zahvalili za slovnični pregled te raziskovalne naloge, profesorici Rosani Breznik pa za pregled in popravo angleškega povzetka v nalogi.

Zahvalili bi se tudi Patricku Koširju za široko paleto znanja o različnih ogrodjih in pomoč pri implementaciji le-teh.

Vsem zgoraj omenjenim posameznikom se še enkrat zahvaljujemo, saj brez Vas raziskovalna naloga ne bi uspela.

Povzetek

V raziskovalni nalogi smo predstavili razvoj aplikacije E-dentiteta, ki zaenkrat omogoča hranjenje dijaških izkaznic in izkaznic ustanov. Na začetku raziskovalne naloge smo postavili različne hipoteze, zatem smo raziskali obstoječe rešitve za hranjenje kartic in pregledali tudi več različnih ogrodij ter knjižnic za postavljanje spletnih aplikacij, med drugim Ionic, Vue, React in Laravel, nato smo se lotili praktičnega dela raziskovalne naloge, kjer smo uspešno postavili aplikacijo za končne uporabnike in aplikacijo za ustanove, ki izdajajo omenjene izkaznice. Med raziskovanjem smo naleteli na nekaj problemov, ki so opisani na koncu raziskovalne naloge, vključno z analizo hipotez, poglavitna problematika naše solucije pa ostaja implementacija hranjenja osebnih izkaznic, saj je varno shranjevanje le-te potreben razvoj lastnega kriptografskega certifikata.

Ključne besede: E-dentiteta, kartice, Ionic ogrodje, Vue.js, elektronske izkaznice.

Abstract

In the research paper, we presented the development of the E-dentiteta application which, as of now, allows users to store student IDs and other institutional cards. At the beginning of the research work, we set various hypotheses. After that, we investigated existing solutions for storing cards on the phone. We also reviewed several different frameworks and libraries for building web applications, including Ionic, Vue, React, and Laravel. Later, we started the practical part of the research work, where we successfully developed an application for end-users and an application meant for institutions, that assign these cards. During our research, we encountered many problems, which are described at the end of the research paper, alongside the hypothesis analysis, but the main problem that our solution presents is the implementation of storing the identity card, because an in-house developed certificate would be needed, I order to securely store the identity card.

Keywords: E-identity, cards, Ionic framework, Vue.js, virtual cards.

Kazalo vsebine

Kazalo slik

Uporabljene kratice

1D – enodimenzionalno

2D – dvodimenzionalno

3D – tridimenzionalno

API – (angl. Application Programming Interface) – vmesnik za namensko programiranje ali aplikacijski programski vmesnik

CSS – (angl. Cascading Style Sheets) – programski jezik za pisanje stilskih predlog

DOM – (angl. Document Object Model) – objektni model dokumenta

EMŠO – Enotna matična številka občana

EU – Evropska unija

GDPR – (angl. General Data Protection Regulation) – Splošna uredba o varstvu podatkov

HTML – (angl. Hyper Text Markup Language) – označevalni programski jezik za programiranje spletnih strani

HTTP – (angl. HyperText Transfer Protocol) – protokol za prenos hiperteksta

iOS – (angl. iPhone OS) – operacijski sistem za telefone Apple

JS – (angl. JavaScript) – programski jezik za programiranje spletnih aplikacij

JSON – (angl. JavaScript Object Notation) – jezik za notacijo objektov

JSX – (angl. JavaScript XML) – razširitev sintakse za JavaScript

JWT – (angl. JSON Web Token) – spletni žeton za JSON

NFC – (angl. Near Field Communication) – tehnologija za komuniciranje v bližini

NPM – (angl. Node Package Manager) – upravitelj paketov za programski jezik JavaScript

PHP – (angl. PHP (Personal Home Page): Hypertext Preprocessor) – programski jezik za programiranje spletnih aplikacij

QR koda – (angl. quick response code) – črtna koda

SHA – (angl. Secure Hash Algorithms) – varni zgoščevalni algoritem

SPA – (angl. Single-Page Application) – aplikacija na eni spletni strani

SSL – (angl. Secure Sockets Layer) – tehnologija za ohranjanje varnosti internetne povezave

TLS – (angl. Transport Layer Security) – protokol za varnost transportne strani

UI – (angl. User interface) – uporabniški vmesnik

ZEKom – Zakon o elektronskih komunikacijah

ZVOP – Zakon o varstvu osebnih podatkov

XML – (angl. Extensible Markup Language) – označevalni jezik za spletne strani

1 Uvod

Pametni telefon je od svojega izuma pa do danes postal nepogrešljiv del vsakdana. Brez njega si današnja družba življenja sploh ne zna predstavljati. Njihova dostopnost in frekvenca uporabe je omogočila, da so preko aplikacij nadomestili marsikatero konvencionalno rešitev. Tako smo, ko je eden izmed članov naše raziskovalne skupine doma pozabil denarnico, in z njo vse svoje identifikacijske elemente, sklenili razviti programsko rešitev, ki bi ustanovam in uporabnikom olajšala uporabo identifikacijskih elementov.

Glavna problematika, ki smo jo želeli rešiti tekom razvoja te aplikacije je bila implementacija sistema za digitalne dijaške izkaznice, saj ni bilo zaslediti nobenega podjetja, ki bi na slovenskem trgu to ponujalo. Seveda pa se pri aplikaciji nismo želeli omejiti le na izkaznice za dijake, zato smo si zamislili pol-odprti sistem dodajanja uporabnikov in izkaznic, saj smo le tako lahko zagotovili, da je uporabnik član neke organizacije, in da ima pri tej organizaciji dodeljeno specifično izkaznico. Več o idejni in aplikacijski zasnovi pa lahko najdete v nadaljevanju raziskovalne naloge.

1.1 Hipoteze

Cilj raziskovalne naloge je bil uspešno zasnovan in delujoč prototip aplikacije za hranjenje izkaznic, ki sestoji iz dveh ločenih delov: uporabniškega in administratorskega.

V raziskovalni nalogi smo postavili naslednje hipoteze:

1) Uporaba ogrodja Laravel bo optimizirala naš zaledni[1] del.
2) Knjižnica React bo najboljša opcija za naše potrebe.
3) Uporaba knjižnice JWT, v kombinaciji s SSL certifikatom, bo dovolj dobra za varnost naše celotne aplikacije na trenutni ravni.

[1] backend

1.2 Metode raziskovanja

Za pripravo raziskovalne naloge smo imeli na razpolago nekaj strokovnih virov in literature na to tematiko, saj je to področje, o katerem se zadnje čase govori vedno več. V oporo so nam bili podobni projekti in določene pretekle raziskovalne naloge v zvezi z e-denarnicami. Večino stvari pa smo morali narediti sami, saj v danem nišnem področju ni obilja podobnih projektov ali pa se ti projekti od naše idejne zasnove preveč razlikujejo, zato jih ni bilo smiselno analizirati.

1.3 Struktura raziskovalnega dela

V raziskovalni nalogi smo najprej predstavili temo in si zadali hipoteze. V drugem poglavju smo predstavili obstoječe rešitve za hranjenje kartic, nato pa smo v tretjem poglavju na hitro predstavili vse zakone, ki jih moramo upoštevati pri gradnji aplikacije. V četrtem poglavju smo predstavili vsa orodja in programe, ki smo jih uporabili za izdelovanje aplikacije. Sam potek praktičnega dela raziskovalne naloge smo predstavili v petem poglavju, analiza in rezultati naše raziskovalne naloge pa so predstavljeni v šestem poglavju.

2 Raziskava obstoječega trga

Na začetku raziskovalne naloge smo morali najprej pregledati trg obstoječih rešitev hranjenja kartic.

Že dolgo nazaj je podjetje Apple oznanilo, da bo razvilo e-denarnico imenovano Apple Passbook. Pred kratkim so jo preimenovali v Apple Wallet in jo dodali v iOS naprave. Med raziskovanjem smo odkrili tudi dve aplikaciji za telefone Android, ki pa sta zelo podobni Apple Wallet.

2.1 Apple Wallet

Apple Wallet je izšel z operacijskim sistemom iOS 6, leta 2012, pod imenom Apple Passbook. S posodobitvijo operacijskega sistema iOS 9 leta 2019 se je aplikacija preimenovala v Apple Wallet. Leta 2019 so Applovi denarnici dodali še možnost dodajanja kreditnih kartic, a le v Združenih državah Amerike. Apple Wallet omogoča prikazovanje 2D in 1D črtnih kod, avtomatsko prižiganje na do 10 izbranih lokacijah, prikaz osebnih dokumentov pri prečkanju državnih meja in še mnogo več. Omogočeno je tudi branje kratic preko NFC funkcije, vendar je ta funkcija dostopna le za namenske izkaznice. (Wikipedia, 2012)

Slika 1: Razlika med 1D in 2D črtnimi kodami

(Vir: prevzeto od (Shopify))

2.2 ID 123

ID 123 je e-denarnica, ki ima tako spletno aplikacijo kot mobilno aplikacijo. Začeli so jo razvijati leta 2017. Aplikacija je mišljena za podjetja, šole in druge ustanove, ki želijo stopiti v korak s časom ter digitalizirati študentske kartice, kartice zaposlenih in druge kartice. Storitev ID123 se začne pri 200 $ na leto za shranjevanje do 50 kartic, cena le-te pa lahko znaša vse do 7500 $ na leto za shranjevanje do 5000 kartic. (ID123)

Slika 2: Cena storitev ID 123

(Vir: prevzeto od (ID123))

2.3 Cards – Mobile Wallet

Cards je aplikacija, ki je najbolj podobna Apple Wallet. Omogoča branje kartic preko NFC senzorja in črtnih kod. V aplikaciji lahko uporabnik shranjuje bančne kartice, kartice zaposlenih, kartice pametnih vrat, zdravstveno kartico, kartico za prevoze, osebno izkaznico, vozniško kartico in še več. (Cards)

3 Zakonodaja

Če smo želeli ustvariti našo aplikacijo, smo se seveda morali vprašati, kaj nam zapovedujejo zakoni na tem področju, kako je z varstvom osebnih podatkov, katere podatke lahko aplikacija sploh zadržuje o uporabnikih in kako mora naša aplikacija komunicirati z našim strežnikom.

3.1 Zakon o elektronskih komunikacijah

Že v letu 2020 bi Slovenija morala posodobiti ZEKom-1 v skladu z novimi direktivami EU. To je Slovenija storila šele slabi dve leti pozneje, ko je bil sprejet ZEKom-2 in to le malo za tem, ko je EU vložila tožbo proti Sloveniji in ji želela izdati denarno kazen. (Služba Vlade Republike Slovenije za digitalno preobrazbo, 2022)

Zaradi tega dejstva smo se odločili, da bomo morali predelati ne le ZEKom-1, ampak tudi ZEKom-2, ker je le-ta bil sprejet za tem, ko smo začeli pisati raziskovalno nalogo.

ZEK-om na splošno ureja zagotavljanje univerzalne storitve, določa pogoje za omejitev lastninske pravice, pravice uporabnikov, ureja varnost omrežij in storitev ter ureja varovanje pravice do komunikacijske zasebnosti uporabnikov javnih komunikacijskih storitev. Zakon je sestavljen iz več zgoraj omenjenih podpoglavij, zato bomo opisali tiste, ki se navezujejo na našo temo. (Služba Vlade Republike Slovenije za zakonodajo)

3.1.1 ZEKom-1

Izvajalec komunikacijskih storitev, v našem primeru mi, moramo zagotoviti varnost svojih storitev. Pri tem mora biti tveganje za nevarnost minimalno. Izvajalec mora zagotoviti, da lahko osebne podatke vidijo le pooblaščene osebe, varovati mora shranjene in ali poslane osebne podatke pred nepooblaščenim ali nezakonitim razkritjem ter izvajati varnostno politiko pri obdelavi osebnih podatkov. Če pride do tveganja za varnost, mora izvajalec po ustreznih komunikacijskih poteh o tem obvestiti vse uporabnike, ki jih tveganje zadeva. Zakon teži k temu, da je maksimalno število podatkov zašifriranih. Zakon se dotakne tudi teme piškotkov in jasno pove, da mora uporabnik privoliti v te, ko je bil predhodno obveščen o upravljalcu ter namenih obdelave podatkov. (Služba Vlade Republike Slovenije za zakonodajo)

3.1.2 ZEKom-2

Zakon ZEKom-2 na potek naše raziskovalne naloge ne vpliva skoraj nič, saj se nanaša bolj na izvajalce javnih komunikacij v Sloveniji. (Ministrstvo za javno upravo)

3.2 Splošna uredba o varstvu podatkov

Pri zakonu GDPR smo ugotovili, da poudarja transparentnost dejanj. Iz zakona GDPR smo zato povzeli nekaj izvlečkov, ki niso zelo pomembni za našo raziskovalno nalogo, vendar se nam zdi, da bi bilo dobro, da jih pozna vsak prebivalec Evropske unije.

Skozi ves GDPR zakon je večkrat omenjeno, da so lahko osebni podatki osebe obdelani le, če se je le-ta strinjala z obdelavo osebnih podatkov za specifičen namen. Izvajalec neke storitve mora imeti dokaz, da se je ta oseba strinjala z obdelavo podatkov. To po navadi organizacije dosežejo preko splošnih pogojev uporabe, ki pa jih skoraj nihče nikoli v celoti ne prebere. Oseba lahko kadarkoli odstopi od pogodbe o sodelovanju o obdelavi podatkov. A ta preklic ne vpliva na obdelavo podatkov, ki se je zgodila med dovoljenjem. Ko pa ta oseba prekliče pogodbo o obdelavi podatkov, se mora vsaka nadaljnja obdelava podatkov končati, hkrati pa se mora izbrisati vse podatke osebe iz sistema. Če je otrok mlajši od 16 let, mora le-ta pridobiti soglasje s strani staršev o obdelovanju osebnih podatkov. Prepovedana je vsaka obdelava osebnih podatkov, ki lahko razkrijejo etično ali rasno poreklo, politično mnenje, versko ali filozofsko prepričanje, posameznikovo spolno življenje ali spolno usmerjenost ter še veliko drugih stvari. Zgoraj omenjene podatke lahko organizacija pridobi, če oseba izrecno privoli v to, ali pa je to potrebno za zaščito življenja ter v primeru, da posameznik to objavi sam. Informacije o kazenskih postopkih se ne smejo hraniti, razen če to odobri organ države ali to le-ta nadzoruje. (Uradni list Evropske unije, 2016)

Pravice oseb so zelo pomembne tudi za organizacije, da le-te vedo, kaj morajo narediti v nekaterih primerih. V naslednjem odstavku bomo zato tudi na kratko opisali pravice oseb.

Upravljalec informacij mora v roku enega meseca od prejema zahtevka posameznika le temu razkriti, kaj se je dogajalo z njegovimi osebnimi podatki. Prav tako mora organizacija posamezniku posredovati podatke o tem, katere osebne podatke organizacija hrani o posamezniku, če le-to ta zahteva. Če ima upravljalec utemeljen razlog, zakaj te zahteve ni uspel rešiti v enem mescu (na primer: kompleksnost zahteve), se lahko ta rok podaljša tudi za dva

dodatna mesca. Upravljalec lahko prav tako zahteva dodatno identifikacijo osebe, ki je vložila zahtevo, če ima upravljalec upravičen dvom v njeno identiteto. (Uradni list Evropske unije, 2016)

Upravljalec je dolžen zagotoviti vsaj kontakt in identiteto osebe, ko pridobi njene podatke. S pomočjo pravne podlage za obdelavo osebnih podatkov mora upravljalec zagovarjati namen, s katerim jih obdeluje. Upravljalec mora osebi zagotoviti informacije, če želi prenesti podatke na novo lokacijo. Prav tako mora pri tem predložiti tudi podatke o varnostnih ukrepih ob prenašanju podatkov. Osebi mora upravljalec zagotoviti podatek o času hrambe osebnih podatkov, jo seznaniti s pravico, da lahko od upravljalca zahteva dostop do osebnih podatkov in popravek ali izbris osebnih podatkov, omejitev obdelave ali pravico do ugovora ter pravico do vložitve pritožbe pri nadzornem organu. Upravljalec lahko spremeni obdelovanje osebnih podatkov, a mora pred spremembo uporabnika o tem obvestiti. (Uradni list Evropske unije, 2016)

Upravljalec mora za vsako obdelavo podatkov imeti napisan namen obdelave, opis kategorije posameznikov, na katere se to nanaša, ter kategorijo uporabnikov, ki jim je bilo to razkrito. Na zahtevo organa mora upravljalec tudi posredovati osebne podatke, če je zahteva utemeljena. Upravljavčeva naloga je tudi psevdonimizacija in šifriranje osebnih podatkov. (Uradni list Evropske unije, 2016)

3.3 Zakon o varstvu osebnih podatkov

3.3.1 ZVOP-1

Zakon o varstvu osebnih podatkov (ZVOP-1) dopolnjuje zakon GDPR za ozemlje Slovenije. Zakon pravi, da se morajo osebni podatki obdelovati zakonito in pošteno. Osebni podatki morajo biti primerni glede na namen, za katerega se zbirajo in obdelujejo. Pri tem mora biti varnost osebnih podatkov zagotovljena vsakemu, ne glede na katerokoli osebno okoliščino. Kot smo že omenili, ta zakon velja samo na ozemlju Republike Slovenije. ZVOP-1 se uporablja za vsa podjetja, ki imajo sedež ali podružnico podjetja na območju Republike Slovenije. Iz tega zakona je izključena samo oprema za prenos osebnih podatkov. (Služba Vlade Republike Slovenije za zakonodajo)

Podatki osebe se lahko obdelujejo le, če le-ta poda osebno privolitev. Namen obdelave osebnih podatkov mora biti določen v zakonu ali drugem pisnem dokumentu, ki se nanaša na obdelovanje osebnih podatkov. V zasebnem sektorju se lahko obdeluje osebne podatke brez privolitve, če je bila z osebo sklenjena pogodba in je potrebno obdelati podatke za pogajanje ali sklenitev pogodbe. (Služba Vlade Republike Slovenije za zakonodajo)

Tako kot pri GDPR, se lahko tudi pri ZVOP-1 osebni podatki obdelujejo brez privolitve, če je to nujno potrebno za varovanje življenja. Tudi občutljivi osebni podatki posameznika se lahko obdelujejo le z izrecnim dovoljenjem, ki je po navadi izraženo v pisni obliki. Ob vsaki spremembi obdelovanja osebnih podatkov mora biti posameznik seznanjen z novim načinom obdelave osebnih podatkov. Podatkov iz kazenskih evidenc ali evidenc prekrškov se ne sme zbirati. (Služba Vlade Republike Slovenije za zakonodajo)

Čeprav se lahko osebni podatki obdelujejo le za določene in zakonite namene, ki so bili določeni, se lahko le-te prekrši, če so podatki posredovani tretji osebi za namene obdelovanja za zgodovinske, statistične ali znanstveno-raziskovalne namene. Ta oseba je primorana ob zaključku obdelave uničiti podatke in sporočiti, kdaj in kako jih je uničila, razen če le-to ne določa zakon. (Služba Vlade Republike Slovenije za zakonodajo)

Osebni podatki morajo biti točni in ažurni. Upravljalec osebnih podatkov lahko preveri točnost osebnih podatkov z vpogledom v osebni dokument ali drugo javno listino posameznika pred vnosom podatkov. (Služba Vlade Republike Slovenije za zakonodajo)

Osebni podatki se lahko hranijo le toliko časa, kot je to potrebno za dosego cilja. Za tem se mora osebne podatke izbrisati, razen če so opredeljeni kot arhivsko gradivo. Če uporabnik zahteva podatek o tem, kaj se je dogajalo z njegovimi osebnimi podatki, mora to dobiti posredovano brez stroškov, razen če ni to določeno drugače. Osebne podatke umrlih posameznikov se lahko posreduje le dedičem prvega ali drugega reda, če ti izkažejo interes. To se ne sme storiti, če je umrla zahteva izrecno prepovedala dostop do svojih podatkov. (Služba Vlade Republike Slovenije za zakonodajo)

Osebni podatki morajo biti zavarovani kolikor se le-to da, in sicer z varovanjem prostorov, opreme in programske opreme. Lahko se tudi vzpostavi biometrijski ukrep za zaščito, če je le-ta pomemben za opravljanje dejavnosti, varnosti premoženja ali varnosti tajnih podatkov. (Služba Vlade Republike Slovenije za zakonodajo)

3.3.2 ZVOP-2

Ker je bil ZVOP-1 leta 2004 sprejet, je v sedanjih časih zelo zastarel (Služba Vlade Republike Slovenije za zakonodajo), zato je vlada predlagala sprejetje zakona ZVOP-2, ki se samo še bolj približa zakonu GDPR. Te spremembe v zakonu za nas ne prinesejo nič posebnega, kar ne bi že pokrival zakon GDPR. (Ministrstvo za javno upravo)

3.4 Zakon o osebni izkaznici

Pri tem zakonu je za nas pomembno samo dejstvo, da lahko za identifikacijo osebe na spletu uporabimo osebni dokument. (Služba Vlade Republike Slovenije)

4 Uporabljene tehnologije

V raziskovalni nalogi smo uporabili kar nekaj tehnologij, s pomočjo katerih nam je uspelo nalogo uspešno končati, zato smo se odločili, da bomo le-te predstavili.

4.1 Adobe XD

Da smo si lažje zamislili obliko naše spletne aplikacije, smo uporabili program Adobe XD. Adobe XD je programska oprema za oblikovanje uporabniškega vmesnika in izdelavo interaktivnih prototipov spletnih ter mobilnih aplikacij. Gre za eno izmed najbolj priljubljenih orodij za oblikovanje uporabniškega vmesnika, ki ga je razvilo podjetje Adobe. (Upwork, 2022)

Adobe XD ima zelo veliko različnih elementov za oblikovanje uporabniškega vmesnika, kot so gumbi, besedilna polja, meniji, slike itd. S pomočjo vseh elementov, ki jih ponuja Adobe XD, se lahko izdeluje interaktivne prototipe spletnih in mobilnih aplikacij, ki omogočajo testiranje uporabniške izkušnje preden aplikacija dejansko zaživi. Prav tako nudi možnost ustvarjanja ilustracij ter različnih grafik, vsebuje pa tudi podporo za oblikovanje virtualne resničnosti. Ta se lahko doseže tako, da pri svojih prototipih uporabite tako imenovane tridimenzionalne predmete (3D predmete). Adobe XD zelo dobro deluje z drugimi orodji in tehnologijami, kot so Photoshop, Illustrator, InVision, Sketch, itd. (Upwork, 2022)

Adobe XD omogoča učinkovit razvoj prototipov, ki olajšajo testiranje uporabniške izkušnje preden aplikacija postane operativna, kar pomaga pri izboljšanju uporabniške izkušnje in zmanjšanju stroškov razvoja. (Upwork, 2022)

Slika 3: Logotip orodja Adobe XD

(Vir: prevzeto od (Wikipedia))

4.2 Ionic ogrodje

Za našo aplikacijo smo uporabili Ionic ogrodje[2], ki je odprtokodna platforma za razvoj mobilnih aplikacij in temelji na tehnologijah HTML, CSS in JavaScript. Čeprav je odprtokodna platforma, je še vedno zelo zanesljiva in varna. Omogoča tudi razvoj aplikacij, ki delujejo na več platformah, vključno z Androidom, iOS in spletnimi brskalniki, kar je bila za nas najboljša izbira. (Hackr.io, 2022)

Ionic vključuje veliko knjižnic in orodij, kar nam je zelo olajšalo razvijanje aplikacije. Ena izmed največjih prednosti Ionic ogrodja je, da omogoča razvoj aplikacij z enotno kodo za več platform, kar omogoča hitrejši in enostavnejši razvoj aplikacije. Poleg tega omogoča tudi enostavno integracijo z drugimi tehnologijami in orodji, kot so React, Firebase in drugi. Hkrati pa omogoča mnogo različnih UI elementov, ki so specifični za dano platformo, kot npr. Google MaterialU, Apple UIKit, ipd. (Hackr.io, 2022)

Na voljo je več hibridnih platform za izdelavo spletnih aplikacij. Ionic je v primerjavi z drugimi ogrodji dosti hitrejši in ima enostavnejši uporabniški vmesnik, s katerim se lahko prihrani ogromno časa. (Hackr.io, 2022)

Ionic je bil prvič predstavljen javnosti leta 2013. Od takrat je izšlo že kar nekaj novejših verzij, ki prinašajo nove prednosti. V raziskovalni nalogi smo uporabili najnovejšo različico Ionic ogrodja, Ionic 6, ki je javnosti na voljo od junija 2022. Pozneje se je to izkazalo kot napačna odločitev, saj so okrnili kar nekaj funkcij, ki so bile zelo uporabne. (Hackr.io, 2022)

Slika 4: Logotip ogrodja Ionic

(Vir: prevzeto od (Wikipedia))

[2] framework

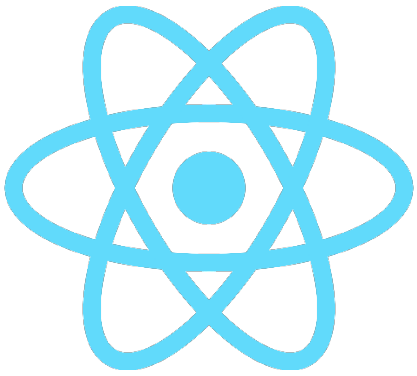
4.3 React.js

Na začetku raziskovalne naloge smo se odločili, da bomo uporabili React.js, vendar se na koncu za le-tega nismo odločili, saj se nam je Vue.js zdel enostavnejši in uporabnejši. React.js je odprtokodna knjižnica za gradnjo uporabniških vmesnikov in spletnih aplikacij v JavaScriptu. React deluje z uporabo koncepta "komponent". Komponenta v Reactu predstavlja del uporabniškega vmesnika, ki ga je mogoče ponovno uporabiti na različnih mestih. Komponente so lahko gnezdene druga v drugi, kar omogoča gradnjo kompleksnejših uporabniških vmesnikov. (Herbert, 2022)

React.js si je zamislil eden izmed Facebookovih programskih inženirjev, saj je želel uporabnikom ponuditi bogatejši in bolj dinamičen uporabniški vmesnik, ki bi bil hitrejši in zmogljivejši. S predstavitvijo knjižnice React se je poenostavil način gradnje uporabniških vmesnikov s komponentami za večkratno uporabo. Kot prvi so React začeli uporabljat ravno pri Facebooku. (Herbert, 2022)

React.js deluje tako, da pri brskalniku zahteva URL naslov spletne strani in komunicira s strežnikom za dostop do strani na spletni strani. Ta vzorec nalaganja spletne strani se nadaljuje za vsako novo spletno stran oz. vir, do katerega je poslana zahteva za dostop. React omogoča uporabnikom, da zgradijo SPA aplikacijo, ki ob zahtevi naloži samo en HTML dokument in posodobi le spremenjen del na spletni strani. (Herbert, 2022)

Zaradi sposobnosti in učinkovitosti razvijanja razširjenih spletnih aplikacij ga danes uporablja na tisoče podjetij, kot so: Facebook, Instagram, Netflix, Uber, Reddit, itd. (Herbert, 2022)



Slika 5: Logotip React.js

(Vir: prevzeto od (Wikipedia))

4.4 Vue.js

Za Vue.js smo se odločili pozneje, saj se nam je zdel enostavnejši za uporabo za namene naše raziskovalne naloge. Vue.js je, od leta 2014, odprtokodno JavaScript ogrodje za gradnjo uporabniških vmesnikov. Prvotno je bil zasnovan kot manjša alternativa obstoječim ogrodjem, vendar se je kmalu izkazal kot zelo zmogljivo ogrodje. (Winter, 2022)

Vue.js se uporablja za gradnjo interaktivnih in dinamičnih uporabniških vmesnikov za spletne aplikacije. Za razliko od drugih ogrodij, kot je Angular in knjižnic, React, se je Vue.js lažje in hitreje naučiti, saj je enostaven in fleksibilen, kar ga naredi idealnega za razvoj manjših in srednje velikih projektov, prav tako je povezljiv z mnogimi drugimi knjižnicami ter ogrodji. (Winter, 2022)

Značilnosti Vue.js je, da vključuje enosmerno vezanje podatkov (angl. one-way data binding), ki omogoča dinamično posodabljanje uporabniškega vmesnika (npr. klepeti v realnem času), z uporabo komponent, ki jih je mogoče ponovno uporabiti. (Winter, 2022)

Vue.js ima tudi veliko skupnost razvijalcev, ki ustvarjajo različne dodatke in orodja za lažje delo z ogrodjem. Med najbolj priljubljenimi orodji za Vue.js so Vuex, Vue Router in Vuetify. (Winter, 2022)

Slika 6: Logotip Vue.js

(Vir: prevzeto od (Wikipedia))

4.5 Razlika med React.js in Vue.js

React.js in Vue.js sta oba odprtokodna JavaScript ogrodja za gradnjo uporabniških vmesnikov, vendar se razlikujeta po nekaterih pomembnih značilnostih.

- Struktura komponent

React.js in Vue.js sta osredotočena na komponento zasnovo, vendar imata različne pristope. V React.js so komponente sestavljene iz ločenih delov za ustvarjanje logike in predstavitve na strani. Vue.js se v tem razlikuje, saj je zasnovan tako, da vključuje logiko in predstavitev v enem delu kode.

- Sintaksa

Vue.js uporablja svojo sintakso, imenovano "Vue Template Syntax", ki omogoča razvijalcem, da pišejo podobno kodo kot pri HTML programiranju. V React.js se uporablja JSX sintaksa, ki omogoča vključevanje JavaScripta v HTML kodo.

- Stanje in upravljanje dogodkov

Vue.js ima sistem za upravljanje stanja, imenovan Vuex, ki je zasnovan za upravljanje velikih in kompleksnih stanj. V React.js se stanje in upravljanje dogodkov lahko izvaja z uporabo knjižnice Redux ali drugih podobnih orodij.

- Učinkovitost

React.js uporablja virtualni DOM, ki je optimiziran za hitrost in učinkovitost pri posodabljanju vmesnikov. Vue.js prav tako uporablja virtualni DOM, vendar ima nekatere lastne funkcije za optimizacijo.

Na splošno je izbira med React.js in Vue.js odvisna od potreb in nalog posameznega projekta. (Herbert, 2022) (Winter, 2022)

4.6 Axios

Axios je odprtokodna JavaScript knjižnica, ki se uporablja za izvajanje HTTP zahtev iz brskalnika ali iz Node.js okolja. Namenjena je poenostavitvi izvajanja HTTP zahtev in zagotavlja intuitivno ter enostavno uporabniško izkušnjo. Omogoča uporabo različnih vrst zahtev, vključno z GET, POST, PUT, DELETE in PATCH. Prav tako omogoča uporabo različnih vrst odgovorov, vključno z JSON in XML. Knjižnica Axios je zasnovana tako, da deluje v več okoljih, vključno z brskalniki in Node.js. Njegova uporaba je enostavna, saj zahteva le nekaj vrstic kode, da se izvede HTTP zahtevo. (The Axios Project)

Slika 7: Logotip Axios

(Vir: prevzeto od (Wikipedia))

4.7 TypeScript

TypeScript je nadgradnja jezika JavaScript, ki zagotavlja dodatno tipizacijo, statično analizo kode in večjo preglednost nad kodo v velikih projektih. TypeScript je odprtokodni jezik, ki ga je razvil Microsoft in je bil prvič izdan leta 2012. Zanj smo se odločili, saj se nam je zdelo, da bo prišel prav pri programiranju spletne aplikacije, saj s tem tvegamo manj napak zaradi nekompatibilnega tipa spremenljivk. (Shubel, 2022)

TypeScript uporablja statično tipizacijo, kar pomeni, da morajo biti tipi podatkov določeni vnaprej. To omogoča preverjanje napak med kodiranjem in lažje razumevanje kode v večjih projektih. TypeScript se lahko prevede v JavaScript in ga je mogoče izvajati v brskalnikih in na strežnikih. Poleg tega podpira tudi številne razvojne okvire in knjižnice, kot so Angular, React, Vue.js in Node.js. (Shubel, 2022)

Prednosti uporabe TypeScripta vključujejo:

- boljšo preglednost kode in večjo učinkovitost pri razvoju v velikih projektih;
- manj napak pri izvajanju kode zaradi statične analize in preverjanja tipov;
- integracijo z obstoječimi orodji in okviri;
- hitro rastoča skupnost in široka podpora.

(Shubel, 2022)

Slika 8: Logotip TypeScript

(Vir: prevzeto od (Wikipedia))

4.8 JSON Web Token

Preden pojasnimo JWT žeton, moramo obrazložiti dva pojma. JSON je kratica za JavaScript Object Notation in je besedilni format za prenos podatkov prek spletnih aplikacij. Žeton je niz podatkov, ki lahko predstavlja različne podatke, na primer identiteto uporabnika. (Akana, 2020)

JSON Web Token je standard za varno prenosljivost podatkov med dvema stranema, ki se uporablja pri spletni avtentifikaciji. JWT je kodirana vrsta podatkov, ki vsebuje informacije o uporabniku in je podpisan z zasebnim ključem. Ta se uporablja največkrat pri API odgovorih. Videti pa je nekako takole: xxxxx.yyyyy.zzzzz. (Akana, 2020)

JWT žeton[3] sestavljajo trije deli: glava[4], podatki [5]in podpis[6]. Glava vsebuje informacije o algoritmu za podpisovanje (vrsto žetona), podatki pa vsebujejo uporabnikove podatke, kot so uporabniško ime, e-poštni naslov ali uporabniško vlogo. Te informacije strežnik običajno uporablja za preverjanje, ali ima uporabnik dovoljenje za izvedbo dejanja, ki ga zahteva. Podpis zagotavlja, da žeton ni bil spremenjen. Stran, ki ustvari JWT, zašifrira tako glavo kot podatke z uporabo zgoščevalne funkcije ali pa ju podpiše z zasebnim ključem. Ta žeton je znan tako izdajatelju kot prejemniku. Ko je žeton uporabljen, prejemnik preveri, ali se glava in podatki ujemajo s podpisom. (Akana, 2020)

Pri tem načinu avtentifikacije strežniku ni treba hraniti baze podatkov z informacijami potrebnimi za identifikacijo uporabnika. Ta omogoča, da se aplikacije in storitve lahko razvijejo, ne da bi bile omejene na uporabo piškotkov ali odvisne od shranjevanja podatkov na strežnik. (Akana, 2020)

[3] token

[4] header

[5] payload

[6] signature

Prednosti JWT vključujejo:

- enostavno implementiranje v različnih programskih jezikih;
- boljšo varnost, saj uporablja simetrično ali asimetrično šifriranje, ki se lahko preveri brez dostopa do podatkovne baze;
- možnost prenosa uporabniških podatkov v šifrirani obliki, kar pomeni, da se uporabnikovih podatkov ne izpostavlja preveč v jasnem besedilu.

Kljub prednostim pa je pomembno opozoriti, da mora biti JWT uporabljen pravilno in skrbno, da se zagotovi varnost in zasebnost uporabnikovih podatkov. (Akana, 2020)

Slika 9: Logotip JWT

(Vir: prevzeto od (Logotip JWT))

4.9 NPM knjižnice

NPM je standardni paketni upravitelj za Node.js. NPM omogoča preprosto namestitev, posodabljanje, odstranjevanje in upravljanje različnih knjižnic, ki jih lahko uporabljamo v projektih. (Abramowski, 2022)

NPM knjižnice so zbirke kode ali modulov v obliki paketov, ki jih lahko uporabljamo v projektih. Ti paketi vsebujejo kodo, dokumentacijo, teste in druge datoteke, ki jih potrebujemo za uporabo določene funkcionalnosti ali knjižnice. (Abramowski, 2022)

NPM knjižnice lahko vsebujejo kodo, ki ponuja različne funkcionalnosti, kot so manipulacija z datotekami, pretvorba datuma in časa, komunikacija z API-ji in še veliko več. Nekatere knjižnice so zelo specializirane in ponujajo funkcionalnost za specifično nalogo, druge pa so bolj splošne in vsestranske. (Abramowski, 2022)

Slika 10: Logotip NPM

(Vir: prevzeto od (Wikipedia))

4.10 PHP

PHP je odprtokodni programski jezik, ki se najpogosteje uporablja za razvoj spletnih aplikacij in dinamičnih spletnih strani. PHP se izvaja na strežniku in omogoča ustvarjanje dinamičnih spletnih vsebin, kot so obrazci, uporabniški profili, forumi, družbena omrežja, spletne trgovine, blogi in drugo. (S., 2023)

PHP je razvil Rasmus Lerdorf leta 1994 kot orodje za spremljanje obiska njegove spletne strani. Od takrat se je PHP močno razširil in postal eden izmed bolj uporabljenih jezikov za spletni razvoj. Danes je podprt z veliko skupnostjo razvijalcev in ima številne knjižnice in orodja, ki olajšujejo razvoj ter vzdrževanje spletnih aplikacij. (S., 2023)

PHP ima preprosto sintakso, ki je podobna C-ju, kar olajša učenje in razvoj aplikacij. Prav tako ima veliko skupnost razvijalcev, ki prispevajo k razvoju orodij in knjižnic, pa tudi k izboljšanju varnosti ter učinkovitosti jezika. PHP je odprtokoden jezik, kar pomeni, da je na voljo brezplačno in da je mogoče prilagoditi njegovo kodo po lastnih potrebah. Lahko se izvaja na večini operacijskih sistemov, kot so Windows, Linux in macOS, pa tudi na strežniških platformah, kot so Apache, Nginx in Microsoft IIS. (S., 2023)

Slika 11: Logotip PHP

(Vir: prevzeto od (Wikipedia))

4.11 Zgoščevanje

Zgoščevanje[7] je proces pretvorbe poljubno dolgega niza bitov v kriptografsko varno kodo poznano tudi kot zgoščena[8] vrednost, katere izhod je vedno sestavljena iz enakega števila bitov, najpogosteje od 128 do 512 bitov. Zgoščevalne funkcije se izkažejo kot izjemno uporabne predvsem pri zgoščevanju velikih datotek, saj lahko uporabnik preveri skladnost neke druge datoteke s svojo lokalno verzijo tako, da primerja njuni zgoščeni vrednosti. Ker so zgoščene vrednosti unikatne za vsak set podatkov, lahko uporabnik z gotovostjo trdi, da ima nespremenjeno kopijo. (Zola, 2021)

4.12 Kriptografija

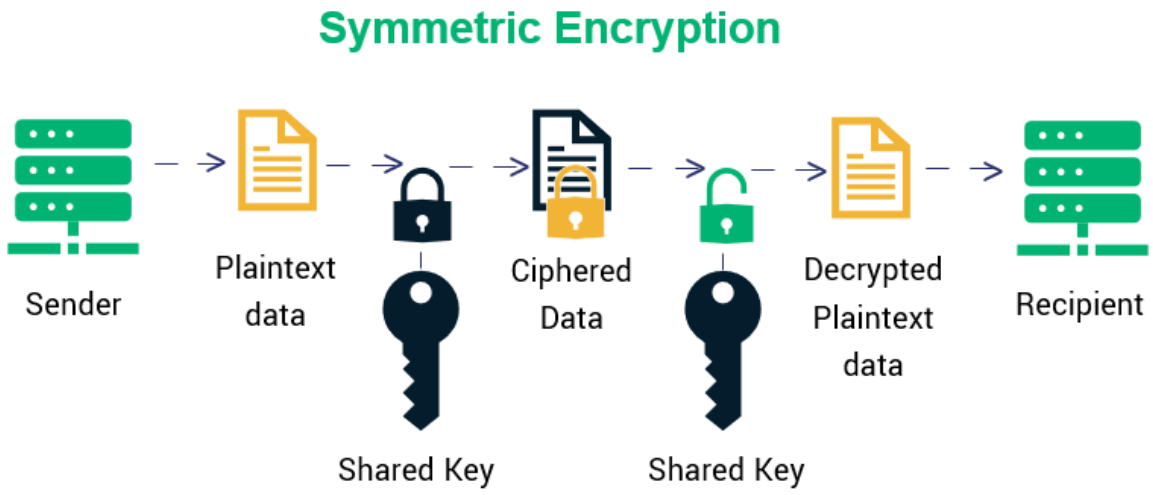
Kriptografija je veda, ki se ukvarja s postopki pretvorbe smiselnih podatkov v šifrirano obliko, s pomočjo matematičnih algoritmov tako, da so le-ti nerazumljivi nepooblaščenim bralcem. Šifriranje se pogosto uporablja za varno pošiljanje in shranjevanje občutljivih podatkov, kot so osebni podatki, finančne informacije ter druge vrste zaupnih informacij. Obstaja več načinov šifriranja podatkov, med drugim simetrično šifriranje in asimetrično šifriranje. Varnostna zaščita podatkov je pomembna za zaščito zasebnosti in preprečevanje kraje identitete ter drugih vrst kibernetskih napadov. (Fruhlinger, 2022)

[7] hashing

[8] hash

4.12.1 Simetrično šifriranje

Simetrično šifriranje uporablja isti ključ za šifriranje kot tudi za dešifriranje podatkov. Ta ključ mora biti dogovorjen preko povezave med pošiljateljem in prejemnikom. Simetrično šifriranje se pogosto uporablja za varno prenašanje podatkov prek interneta, na primer preko protokola SSL/TLS. (Fruhlinger, 2022)

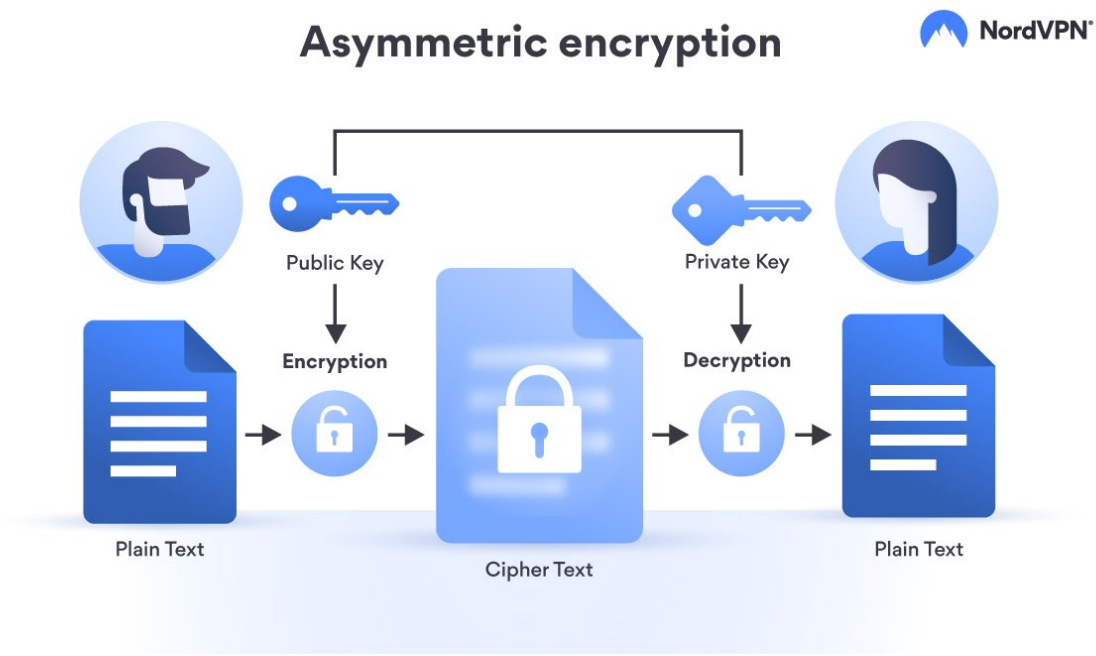


Slika 12: Simetrično šifriranje

(Vir: prevzeto od (SectigoStore))

4.12.2 Asimetrično šifriranje

Asimetrično šifriranje, imenovano tudi šifriranje s pomočjo javnega ključa, uporablja različna ključa za šifriranje in dešifriranje podatkov. Eden izmed ključev se imenuje javni ključ, ki je na voljo vsem, drugi pa se imenuje zasebni ključ, ki ga hrani le lastnik. Asimetrično šifriranje se pogosto uporablja za preverjanje pristnosti in integritete podatkov, ki se prenašajo med prejemnikom in pošiljateljem. (Fruhlinger, 2022)



Slika 13: Asimetrično šifriranje

(Vir: prevzeto od (Nord VPN))

5 Praktični del raziskovalne naloge

5.1 Idejna zasnova

Sama problematika, za katero smo želeli razviti aplikaciji, se je izkazala za zelo kompleksno, zato smo se razvoja aplikacij lotili sistematsko. Pri idejni zasnovi, smo želeli sprva hraniti identifikacijske izkaznice vseh vrst, kar bi pomenilo implementacijo osebnih dokumentov, dijaških oz. študentskih izkaznic ter ostalih izkaznic specifičnih ustanov. Pri razvoju ideje se je težava pojavila pri hranjenju osebnega dokumenta, saj je z omejenim dostopom do podatkov o osebi praktično nemogoče določiti, ali bi bila osebna izkaznica, ki jo je uporabnik dodal v aplikacijo, lahko veljavna. Ta problem smo želeli rešiti z razvojem pol-odprte aplikacijske strukture, ki bi zahtevala, da uporabniški račun za končnega uporabnika ustvari administrator sistema ustanove, pri kateri je uporabnik član. To bi v praksi dodalo človeški faktor avtentifikacije in onemogočilo vnos lažnih informacij v sistem. Implementacija takšnega sistema bi sicer res omogočila kredibilnost informacij o uporabnikih, vendar pa bi bilo potrebno dodati še eno varovalo, ki bi onemogočalo ponarejanje izkaznic specifične ustanove. To varovalo bi lahko predstavljal digitalni certifikat, ki se ga uporabniku dodeli ob kreaciji uporabniškega računa. Takšen certifikat bi se nato uporabljal za podpisovanje izkaznic, ki jih administrator dodeli uporabniku. Implementacija certifikata bi tako v teoriji lahko omogočila implementacijo hranjenja osebnih izkaznic, vse dokler bi administrator sistema lahko ob vnosu podatkov o osebni izkaznici preveril tudi podatke, zapisane na fizični izkaznici (datum poteka, izdajatelj, EMŠO, številka osebnega dokumenta). Veljavnost certifikata bi tako lahko postala vezana na veljavnost osebnega dokumenta, vsaka nova izkaznica podpisana s tem certifikatom, pa bi lahko bila preverljiva brez predložitve fizične osebne izkaznice.

5.2 Problem implementacije idejne zasnove

Čeprav bi zgoraj omenjena idejna zasnova omogočala varno in enostavno validacijo izkaznic, je njena implementacija na trenutni ravni praktično nemogoča, saj bi bilo za delovanje takšnega

sistema potrebno popisovanje osebnih izkaznic vseh uporabnikov, ki bi želeli uporabljati vse funkcije tega produkta. Tako smo bili primorani, če smo želeli aplikaciji dodati tudi uporabno vrednost, našo idejno zasnovo nekoliko prilagoditi. Sistem pol-odprte arhitekture je sicer ostal, spremenil pa se je obseg podatkov in njihova uporabnost, za namen preverjanja veljavnosti identifikacijskih elementov. Za potrebe raziskave, smo si zamislili sistem za učinkovito digitalizacijo dijaških izkaznic in preverjanje le-teh. S tem smo omogočili zbiranje neprecenljivih podatkov o delovanju samega sistema in pripravljenosti uporabnikov za uporabo takšnega sistema, prav tako pa smo želeli ugotoviti, če je naša organizacija (v našem primeru šola) pripravljena takšen sistem ponuditi svojim članom (v našem primeru dijakom). V času nastajanja te raziskovalne naloge se beta testiranje, navedeno zgoraj, še ni pričelo.

5.3 Razvoj aplikacij

S pomočjo prej opisanih orodji smo začeli razvijati aplikaciji. Na začetku smo uporabili orodje Adobe XD, s katerim smo zasnovali prvi uporabniški vmesnik aplikacije. V tem orodju smo le poustvarili koncept aplikacije, torej kako bo aplikacija izgledala in kakšne bodo njene funkcije.

Slika 14: Prvi koncepti E-dentitete

Ko smo se le zadovoljili s konceptom aplikacije, smo se začeli spoznavati z Ionic ogrodjem, saj se nam je le-ta zdel najboljša možnost, saj ponuja veliko različnih opcij za izgradnjo aplikacije med drugim tudi opcijo grajenja aplikacij specifično za dano platformo[9]. Ionic ogrodje ponuja tri različna ogrodja za ustvarjanje interaktivnosti aplikacije: React, Angular in Vue. Najboljša od teh treh možnosti se nam je zdela na začetku uporaba ogrodja React. React nam je predstavljal še neraziskane vode, saj ga nismo uporabnili še pri nobenem od preteklih projetktov. Po nekaj urah intenzivnega učeja smo ugotovili, da je bila izbira ogrodja React napaka, saj je bil preveč zahteven za uporabo in učenje. Tako smo, po nekaj demo prikazih, za logiko aplikacije uporabili Vue.js. Po nekaj dneh smo ustvarili prvi delujoči prototip aplikacije, ki je imel dve skupini zavihkov. V prvi skupini zavihkov smo lahko preklapljali med vpisno stranjo in stranjo za registracijo. V drugi skupini zavihkov pa smo lahko preklapljali med nastavitvami, domačim zavihkom, zavihkom za dodajanje kartic, zavihkom s karticami in zavihkom za izpis iz aplikacije.

Slika 15: Uspešna kreacija domače strani

[9] native build

Kmalu smo usvojili veščino usmerjanja po aplikaciji in veščino zaklepanja zavihkov pred nekim dogodkom. Zaklepanje zavihkov je bilo težje delo, a smo ga z malo truda kmalu usvojili. S tem je bil čelni[10] del aplikacije skoraj končan. Le temu je bilo treba dodati še oblikovanje.

Zatem smo se lotili zaledja[11] naše aplikacije, saj smo lahko le tako zagotovili pravilno delovanje in dejansko funkcionalnost aplikacij. To je bil večji izziv, saj se pred začetkom tega projekta še nismo srečali z razvojem kompleksnejših zalednih struktur. Za pravilno delovanje aplikacij je bil namreč potreben robusten API, ki bi lahko zagotavljal vse podatke, ki jih zahteva čelni del aplikacij. Za začetek smo uporabili ogrodje PHP jezika imenovano Laravel. Le tega smo usvojili v kakšnem tednu. A ko smo ga poskusili implementirati z lokalnega strežnika na strežnik našega ponudnika strežniških storitev, je prišlo do zapletov. Čeprav smo poizkušali večkrat namestiti ogrodje na strežnik, naše funkcije, ki so prej delovale, niso želele delovati. Kontaktirali smo tudi ponudnika, ki nam je posredoval vso uporabno dokumentacijo in nasvete, a nič, kar smo uporabili, ni delovalo. Zato smo naše zaledje aplikacije premaknili kar na sam PHP programski jezik. Tudi tu ni nič teklo po našem primarnem načrtu.

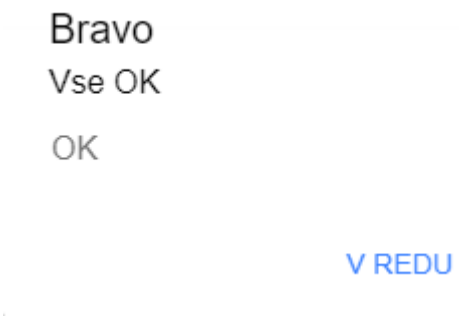
Ko smo vzpostavili preprosti API, ki bi prejel podatke za vpis in jih primerjal z vpisi v podatkovni bazi, le-ta ni deloval. Povezavo smo poskušali vzpostaviti z več knjižnicami, kot so Axios in Fetch API. Poizkusili smo tudi na lokalnem strežniku, a zaradi nekega razloga skripta tudi tam ni delovala. Sprva smo menili, da je problem pošiljanje podatkov preko interneta. Na koncu smo poskusili tudi pošiljanje podatkov z uporabo orodja Postman API. Z uporabo le-tega smo prvič dobili nazaj pozitiven odgovor strežnika, zato smo začeli raziskovati, kaj je bil problem s knjižnicami za pošiljanje podatkov. Eden od prijateljev je predlagal, naj uporabimo namesto ukaza $_POST raje ukaz $_REQUEST. Ker nismo imeli česa izgubiti, smo le-to implementirali v kodo in rezultati so bili presenetljivi.. Naš API začel delovati pravilno. Pri ukazu $_REQUEST pa je bil le en problem, saj se je odgovor strežnika gibal od 100 ms pa tudi do 7 s. Zato smo spet začeli raziskovati, zakaj ukaz $_POST ne deluje. Preizkusili smo tudi predloge različnih blogov, ki so trdili, da bi morali pred seznamom podatkov, ki jih pošiljamo s pomočjo Axiosa, dodati besedo data. Tudi to ni delovalo. Nato pa smo nek večer zasledili

[10] frontend

[11] backend

ukaz v PHP imenovan file_get_contents. Tudi ta lahko, tako kot ukaz $_POST, prebira podatke iz poslane zahteve.

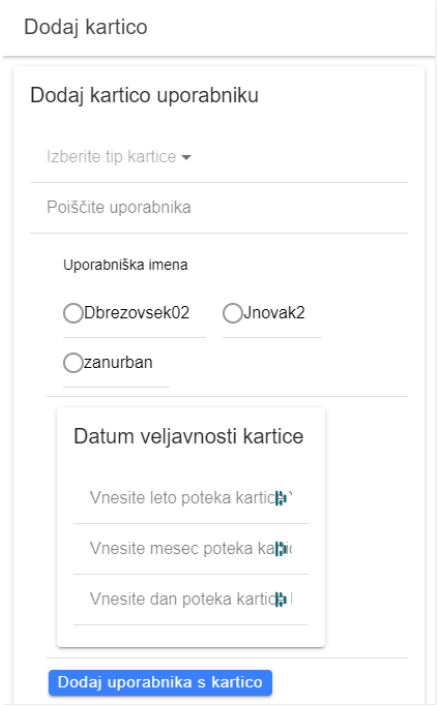
Prvič smo se prebili malo dlje, saj se je API začel odzivati pravilno in hitro. Za skripto za nadzorovanje prijav smo ustvarili še skripte za registracijo, prikaz kartic in prikaz nastavitev.



Slika 16: Odziv API-ja na zahtevo za prijavo

Nato smo se lotili drugega dela aplikacije, s katerim bi lahko preverili veljavnost kartic dodanih v aplikacijo. Ker smo želeli, da bi bil način preverjanja kartic čim preprostejši in uporaben, smo zasnovali dva načina preverjanja kartic. Prvi način bi bil za vsakdanjo uporabo, saj bi lahko črtno kodo preprosto prebral kar z aplikacijo za skeniranje črtnih kod. Pri tem načinu bi lahko uporabnik videl, ali je kartica veljavna ali ne. Seveda pa smo takšno preverjanje ustrezno zaščitili, da ne bi prišlo do poneverbe in ponovne uporabe enakih črtnih kod. Drug način preverjanja pa bi bil preko administratorske aplikacije, s katero bi lahko preveril tudi druge podatke kartice. Ko smo dobili to idejo, smo administratorski aplikaciji dodali več funkcij, in sicer:

- dodajanje kartic uporabnikom,
- dodajanje uporabnikov,
- dodajanje novega tipa kartic.



Slika 17: Dodajanje kartic uporabnikom

Ko so vse funkcije delovale na obeh aplikacijah, smo začeli implementirati JWT žetone. Pri tem smo naleteli na kar nekaj ovir. Ena izmed ovir, ki bi jo izpostavili, je zastarela dokumentacija za JWT žetona. V dokumentaciji smo namreč zasledili, da lahko preverimo integriteto žetona s pomočjo funkcije verify. Ko smo le-to poizkusili, ta ni delovala, saj se nam je izpisala napaka, da je nemogoče klicati skrito metodo. Zato smo začeli brskati po vsevednem internetu in na enem izmed novejših forumov zasledili enak problem. Tu je pisalo, da je po novem funkcija verify že vgrajena v funkcijo decode.

Druga ovira, na katero smo naleteli, pa je bila pošiljanje žetona preko glave zahteve. Tu smo naleteli na težavo, da strežnik le-te ni prejel ob pošiljanju zahteve. Kmalu smo zasledili, da je bil problem v nastavitvah strežnika, kjer je bilo onemogočeno pošiljanje kakršnekoli glave.

JWT žetoni so pomemben del varnosti naših aplikacij in njunih integritet, saj se s podatki, ki so zapisani v žetonih, onemogoča poneverba črtnih kod za validacijo izkaznic, onemogoča se izvajanje klicev na strežnik, če se žeton ne ujema s spremenljivko, shranjeno v seji strežnika, prav tako pa žetoni skrbijo za varno in dokončno odjavo iz sistema ob neaktivnosti ali namenski odjavi.

6 Predstavitev rezultatov raziskovalne naloge

Našo prvo hipotezo smo delno potrdili, saj je bila izbira ogrodja Laravel korak v pravo smer glede optimizacije zalednega dela, saj ima že veliko pred-vgrajenih funkcij za lažje rokovanje s čelnim delom, vendar je na žalost nismo mogli potrditi v celoti, saj smo imeli težave z implementacijo ogrodja na spletni strežnik našega strežniškega ponudnika.

Drugo hipotezo smo morali ovreči, saj se je na koncu izkazalo, da je bila implementacija ogrodja Vue v našo aplikacijo boljša odločitev, saj je to ogrodje imelo bolje prilagojene funkcije, ki smo jih potrebovali v naši raziskovalni nalogi kot pa knjižnica React.

Zadnjo hipotezo pa smo lahko potrdili, saj se je uporaba knjižnice JWT izkazala za dobro prakso. Če bi nekdo želel prestreči podatke brez žetona, bi bilo to zelo lahko, saj ni unikatnega identifikatorja, s katerim bi strežnik lahko z gotovostjo trdil, da na drugi strani komunicira s pravim klientom. Po implementaciji JWT žetona pa to ni mogoče, saj se žeton generira ob prijavi in je uporabljen v vsaki nadaljnji zahtevi. Če se žeton prestreže, je le-ta neuporaben, saj dejanski uporabnik že uporablja žeton za izvajanje zathev, prav tako pa se spremenljivka seje te tretje osebe ne bi ujemala s podatki seje, shranjenimi na strežniku.

7 Zaključek

Med pisanjem raziskovalne naloge smo prišli do veliko novega znanja.

Poglobili smo naše znanje o programiranju tako čelnega kot zalednega sistema ter izvedeli veliko novega o ogrodjih in knjižnicah, ki se uporabljajo pri postavitvi spletnih storitev. Na poti smo naleteli na mnogo težav – od iskanja in usposabljanja ogrodij do dizajna svoje aplikacije.

Začeli smo se učiti o knjižnici React, ki se nam je po nekaj urah intenzivnega učenja zdela, za naše pojme in potrebe, korak v napačno smer. Zaradi le tega smo bili prisiljeni v iskanje alternative. Po nekaj dneh smo naleteli na ogrodje Vue, ki se nam je zdel zelo enostaven in uporaben za naše namene, saj se je pisal zelo podobno kot HTML koda, ki pa smo jo med našim šolanjem že večkrat pisali.

Tudi pri zalednem sistemu smo morali spremeniti veliko stvari, ki prvotno niso bile mišljene. Za začetek smo začeli zaledje sistema vzpostavljati v Laravel ogrodju za programski jezik PHP. Kmalu smo postavili preprost API in ga želeli tudi preizkusiti na strežniku ter ne samo na lokalnem strežniku. Ko smo le-to poizkusili, nam ni uspelo, saj sta bila iz neznanega razloga ogrodje in strežnik nekompatibilna.

Med razvojem spletne aplikacije smo prišli tudi do ideje, da bi razvili še aplikacijo, s katero bi lahko preveril verodostojnost vnesenih kartic. V ta namen smo se začeli spoznavati z NPM knjižnico, ki omogoča nameščanje različnih knjižnic, ki jih je razvila skupnost. Tako smo naleteli na knjižnico, s katero lahko generiramo in preberemo črtne kode. V ta namen smo administratorski aplikaciji dodali funkcije upravljanja z uporabniki in njihovimi izkaznicami, prav tako imajo le administratorji možnosti dodajanja novih uporabnikov, ustvarjanja novih izkaznic ter zmožnost dodeljevanja izkaznic uporabnikom. To v praksi omogoča, da so vse izkaznice, ki jih uporabnik lasti, veljavne in imajo verodostojne podatke.

Letošnja raziskovalna naloga je le del večjega, obširnejšega projekta, ki bo trajal več let. V bližnji prihodnosti želimo še povečati varnost te aplikacije, prav tako pa želimo omogočiti uporabo osebnih izkaznic in nanje vezanih digitalnih certifikatov za podpisovanje in dokazovanje verodostojnosti in veljavnosti izkaznic.

VIRI IN LITERATURA

Abramowski, N. (28. november 2022). What is NPM? A Beginner's Guide. Pridobljeno 3. februar 2023 iz https://careerfoundry.com/en/blog/web-development/what-is-npm/

Akana. (8. december 2020). What Is JWT? Pridobljeno 3. februar 2023 iz https://www.akana.com/blog/what-is-jwt

Cards. (brez datuma). Cards Developers. Pridobljeno 16. september 2022 iz https://www.cards.app/Developers/

Fruhlinger, J. (22. maj 2022). What is cryptography? How algorithms keep information secret and safe. Pridobljeno 20. februar 2023 iz https://www.csoonline.com/article/3583976/what-is-cryptography-how-algorithms-keep-information-secret-and-safe.html

Hackr.io. (15. december 2022). What is Ionic Framework? and Why Use it over Other Frameworks? Pridobljeno 12. januar 2023 iz https://hackr.io/blog/ionic-framework

Herbert, D. (27. julij 2022). What is React.js? (Uses, Examples, & More). Pridobljeno 20. januar 2023 iz https://blog.hubspot.com/website/react-js

ID123. (brez datuma). ID123. Pridobljeno 16. september 2022 iz https://www.id123.io/

ID123. (brez datuma). ID123 Pricing. Pridobljeno 16. september 2022 iz https://www.id123.io/pricing/

Logotip JWT. (brez datuma). Pridobljeno 6. marec 2023 iz https://miro.medium.com/v2/resize:fit:788/1*XkmnsJ6Joa6EDFVGUw0tfA.png

Ministrstvo za javno upravo. (brez datuma). ZAKON O ELEKTRONSKIH KOMUNIKACIJAH. Pridobljeno 18. september 2022 iz https://e-uprava.gov.si/drzava-in-druzba/e-demokracija/predlogi-predpisov/predlog-predpisa.html?id=10097

Ministrstvo za javno upravo. (brez datuma). ZAKON O VARSTVU OSEBNIH PODATKOV. Pridobljeno 28. oktober 2022 iz https://e-uprava.gov.si/drzava-in-druzba/e-demokracija/predlogi-predpisov/predlog-predpisa.html?id=10208

Nord VPN. (brez datuma). Asimetrično šifriranje. Pridobljeno 6. marec 2023 iz

S., A. (1. marec 2023). What Is PHP? Learning All About the Scripting Language. Pridobljeno 2. marec 2023 iz https://www.hostinger.com/tutorials/what-is-php/

SectigoStore. (brez datuma). Simetrično šifriranje. Pridobljeno 6. marec 2023 iz

Shopify. (brez datuma). 1D VS 2D BARCODE SCANNER. Pridobljeno 16. septemebr 2022 iz https://cdn.shopify.com/s/files/1/0537/6001/files/barcodes_600x600.jpg?v=1563855063

Shubel, M. (26. julij 2022). What Is TypeScript? Pridobljeno 3. februar 2023 iz https://thenewstack.io/what-is-typescript/

Služba Vlade Republike Slovenije za digitalno preobrazbo. (14. september 2022). Na matičnem delovnem telesu Državnega zbora sprejeti amandmaji k predlogu Zakona o elektronskih komunikacijah (ZEKom-2). Pridobljeno 18. september 2022 iz https://www.gov.si/novice/2022-09-14-na-maticnem-delovnem-telesu-drzavnega-zbora-sprejeti-amandmaji-k-predlogu-zakona-o-elektronskih-komunikacijah-zekom-2/

Služba Vlade Republike Slovenije za zakonodajo. (brez datuma). Zakon o elektronskih komunikacijah (ZEKom-1). Pridobljeno 18. september 2022

Služba Vlade Republike Slovenije za zakonodajo. (brez datuma). Zakon o varstvu osebnih podatkov (ZVOP-1). Pridobljeno 28. oktober 2022 iz http://pisrs.si/Pis.web/pregledPredpisa?id=ZAKO3906

Služba Vlade Republike Slovenije. (brez datuma). Zakon o osebni izkaznici (ZOIzk-1). Pridobljeno 10. november 2022 iz http://www.pisrs.si/Pis.web/pregledPredpisa?id=ZAKO5758

The Axios Project. (brez datuma). Axios - Getting started. Pridobljeno 22. januar 2023 iz https://axios-http.com/docs/intro

Upwork. (29. junij 2022). What Is Adobe XD? Top Uses, Features, and Applications. Pridobljeno 20. januar 2023 iz https://www.upwork.com/resources/what-is-adobe-xd

Uradni list Evropske unije. (4. maj 2016). UREDBA (EU) 2016/679 EVROPSKEGA PARLAMENTA IN SVETA z dne 27. aprila 2016 o varstvu posameznikov pri obdelavi osebnih podatkov in o prostem pretoku takih podatkov ter o razveljavitvi Direktive 95/46/ES (Splošna uredba o varstvu podatkov). Pridobljeno 16. september 2022 iz https://eur-lex.europa.eu/legal-content/SL/TXT/PDF/?uri=CELEX:32016R0679&from=SL

Wikipedia. (22. september 2012). Apple Wallet. Pridobljeno 16. september 2022 iz https://en.wikipedia.org/wiki/Apple_Wallet

Wikipedia. (brez datuma). Logotip Adbobe XD. Pridobljeno 6. marec 2023 iz https://upload.wikimedia.org/wikipedia/commons/thumb/c/c2/Adobe_XD_CC_icon.svg/1200px-Adobe_XD_CC_icon.svg.png

Wikipedia. (brez datuma). Logotip Axios. Pridobljeno 6. marec 2023 iz https://upload.wikimedia.org/wikipedia/commons/thumb/d/d1/Axios_%28computer_library%29_logo.svg/1280px-Axios_%28computer_library%29_logo.svg.png

Wikipedia. (brez datuma). Logotip Ionic. Pridobljeno 6. marec 2023 iz https://upload.wikimedia.org/wikipedia/commons/thumb/d/d1/Ionic_Logo.svg/1280px-Ionic_Logo.svg.png

Wikipedia. (brez datuma). Logotip NPM. Pridobljeno 6. marec 2023 iz

Wikipedia. (brez datuma). Logotip PHP. Pridobljeno 6. marec 2023 iz

Wikipedia. (brez datuma). Logotip React.js. Pridobljeno 6. marec 2023 iz https://upload.wikimedia.org/wikipedia/commons/thumb/a/a7/React-icon.svg/2300px-React-icon.svg.png

Wikipedia. (brez datuma). Logotip TypeScript. Pridobljeno 6. marec 2023 iz https://upload.wikimedia.org/wikipedia/commons/thumb/4/4c/Typescript_logo_2020.svg/1200px-Typescript_logo_2020.svg.png

Wikipedia. (brez datuma). Logotip Vue.js. Pridobljeno 6. marec 2023 iz https://upload.wikimedia.org/wikipedia/commons/thumb/9/95/Vue.js_Logo_2.svg/1184px-Vue.js_Logo_2.svg.png

Winter, R. (18. julij 2022). What Is Vue.js? (Uses + 5 Website Examples). Pridobljeno 20. januar 2023 iz https://blog.hubspot.com/website/vue-js

Zola, A. (junij 2021). DEFINITION hashing. Pridobljeno 3. februar 2023 iz https://www.techtarget.com/searchdatamanagement/definition/hashing