Istraživanje, razvoj i analiza različitih metoda autentikacije u web aplikacijama

Trelec, Jakob

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni studij

ISTRAŽIVANJE, RAZVOJ I ANALIZA RAZLIČITIH METODA AUTENTIKACIJE U WEB APLIKACIJAMA

Završni rad

Jakob Trelec

Osijek, 2024.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

Obrazac Z1P: Obrazac za ocjenu završnog rada na sveučilišnom prijediplomskom studiju

Ocjena završnog rada na sveučilišnom prijediplomskom studiju

Ime i prezime pristupnika:	Jakob Trelec
Studij, smjer:	Sveučilišni prijediplomski studij Računarstvo
Mat. br. pristupnika, god.	R4579, 28.07.2020.
JMBAG:	0165086887
Mentor:	izv. prof. dr. sc. Ivica Lukić
Sumentor:	
Sumentor iz tvrtke:	
Naslov završnog rada:	Istraživanje, razvoj i analiza različitih metoda autentikacije u web aplikacijama
Znanstvena grana završnog rada:	**Informacijski sustavi (zn. polje računarstvo)**
Zadatak završnog rada:	Istražiti različite metode autentikacije uključuju osnovnu prijavu s korisničkim imenom i lozinkom, koja zahtijeva siguran način pohrane lozinki. Također, tu je OAuth 2.0 za integraciju s društvenim mrežama i postojećim računima, Firebase Authentication za brzu autentikaciju, te OpenID Connect za povezivanje s različitim identitetskim izvorima. Multi-Factor Authentication (MFA) i biometrijska autentikacija povećavaju sigurnost s više faktora i biometrijskim podacima, dok se autentikacija putem kripto novčanika koristi u decentraliziranim aplikacijama. Specifičnije metode uključuju SAML, Kerberos, JWT,

Datum prijedloga ocjene završnog rada od strane mentora:	16.09.2024.
Prijedlog ocjene završnog rada od strane mentora:	Izvrstan (5)
Datum potvrde ocjene završnog rada od strane Odbora:	25.09.2024.
Ocjena završnog rada nakon obrane:	Izvrstan (5)
Datum potvrde mentora o predaji konačne verzije završnog rada čime je pristupnik završio sveučilišni prijediplomski studij:	26.09.2024.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O IZVORNOSTI RADA

Osijek, 26.09.2024.

Ime i prezime Pristupnika:	Jakob Trelec
Studij:	Sveučilišni prijediplomski studij Računarstvo
Mat. br. Pristupnika, godina upisa:	R4579, 28.07.2020.
Turnitin podudaranje [%]:	10

Ovom izjavom izjavljujem da je rad pod nazivom**: Istraživanje, razvoj i analiza različitih metoda autentikacije u web aplikacijama**

izrađen pod vodstvom mentora izv. prof. dr. sc. Ivica Lukić

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis pristupnika:

SADRŽAJ

1. UVOD

U ovom diplomskom radu obrađuje se tema istraživanje, razvoj i analiza različitih metoda autentikacije u web aplikacijama. Autentikacija, kao ključni dio kibernetičke sigurnosti, osigurava da su korisnici, procesi i uređaji koji pristupaju aplikacijama ovlašteni za takvu aktivnost. U radu će se ispitati razne metode autentikacije od tradicionalnih metoda pristupa kao što su korisničko ime i lozinka, do modernijih kao što su dvofaktorske autentikacije (*engl. Two-Factor Authentication, 2FA*), autentikacije putem e-pošte ili društvenih mreža, tokena, novčanika za kripto valute. Cilj je ocijeniti njihovu učinkovitost, lakoću korištenja te otpornost na sigurnosne prijetnje.

Konstantnim razvojem tehnologije web aplikacije postale su dio svakodnevnog života te je velik udio dokumenata na internetu prikazan upravo preko web aplikacija. Web aplikacije koriste se kako bi olakšale prikaz i rad sa web dokumentima, omogućile su lakšu komunikaciju, brži pristup velikoj količini informacija, Internet trgovanje, rad na daljinu, Internet bankarstvo, modernizirale su zdravstvo i još mnogo drugih mogućnosti. Web aplikacije postale su esencijalni alati za osobnu, poslovnu i društvenu interakciju te je osiguranje integriteta i povjerljivosti korisničkih podataka od velike važnosti.

Rad je podijeljen na pet poglavlja. Uvodno poglavlje objašnjava zadatak diplomskog rada. Drugo poglavlje daje pregled u područje teme. Treće poglavlje pruža teorijski opis autentikacije i način na koji se ostvaruje u web aplikacijama. Četvrto poglavlje opisuje kreirano okruženje, korištene tehnologije pri njegovom razvoju te samu implementaciju rješenja. U petom poglavlju donosi se zaključak na temelju raznih implementiranih i ispitanih metoda autentikacije.

1.1 Zadatak završnog rada

Istražiti i analizirati različite metode autentikacije, prijavu s lozinkom, OAuth 2.0, MFA i kripto novčanike.

2. PREGLED PODRUČJA TEME

2.1 Razvoj autentikacije

Eksponencijalnim razvojem interneta te njegovim svakodnevnim korištenjem došla je u pitanje sigurnost i potvrda identiteta na internetu. Cilj autentikacije je potvrđivanje pojedinca ili sustava da su ono za što se predstavljaju te da samo ovlašteni pojedinci ili sustavi imaju pristup određenom resursu. Potreba za autentikacijom pojavila se ranih 1960-ih godina kada je Fernando Corbató(1926 – 2019, američki informatičar) uveo sustav lozinki na računalo koje je koristilo više ljudi kako bi omogućio zaštitu njihovih datoteka od neovlaštenog pristupa. Lozinke su se čuvale na računalu u obliku teksta što je i dalje predstavljalo problem jer su dugi korisnici mogli pronaći datoteku u kojoj su spremljene lozinke te tako svejedno imati pristup tuđim datotekama što je dovelo do uvođenja enkripcije pri spremanju lozinki, 1974. Robert Morris(1932 – 2011, američki kriptograf) predlaže korištenje hash funkcija pri pohranjivanju lozinki koje se koriste i dan danas [1]. Ranih 2000-ih godina programeri su razvili snažnije tehnologije autentikacije koje su zahtijevale višeslojne potvrde identiteta odnosno višefaktorska autentikacija (engl. *Multi-Factor authentication, MFA*), kasnijih 2000-ih godina daljnjim razvojem tehnologije i potrebom za višom sigurnošću kreću se upotrebljavati biometrijske metode autenticiranja koje su se prvotno koristile za pristup u državne sektore visoke sigurnosti te u vojne svrhe [2]. Pojavom pametnih telefona biometrijska autentikacija postala je dio svakodnevice. Autentikacija funkcionira na način da korisnik upisuje svoj jedinstveni nadimak, e-poštu i sl. te svoju lozinku koji se zatim uspoređuju sa onima u bazi podataka ovlaštenih korisnika, baza podataka može se nalaziti na poslužitelju lokalnog operativnog sustava ili na udaljenom poslužitelju, ako se nadimak i lozinka podudaraju sa onima u bazi korisnik postaje ovlašten za određeni resurs te ga može koristiti ili ima pristup nekim pravima povezanim s korisnikom [3].

2.2 Kategorizacija i metode autentikacije

Autentikacija je podijeljena u tri kategorije: autentikacija s jednim faktorom (engl. *Single-Factor Authentication, SFA*), dvofaktorska i višefaktorska autentikacija . Autentikacija s jednim faktorom najjednostavnija je i najzastupljenija, korisnik mora potvrditi identitet samo jednim faktorom odnosno nadimkom i lozinkom, dvofaktorska autentikacija funkcionira slično kao jednofaktorska ali uz dodatan faktor potvrde identiteta kao što su na primjer dodatne lozinke od par znakova ili brojeva koji se pošalju korisniku SMS-om (*engl. Short Message Service*), e-‚poštom ili nekom drugom sličnom metodom. Na posljetku, višefaktorska autentikacija je metoda provjere u više faktora, najkompleksnija je te zahtjeva od korisnika više različitih

čimbenika za provjeru identiteta, ali zato pruža najvišu razinu sigurnosti. Koristi istovremeno više faktora autenticiranja kako bi potvrdila korisnikov identitet, faktor potvrde mogu biti: faktor znanja kao što su lozinka, nadimak, sigurnosno pitanje i slično; faktor posjeda odnosno nešto što korisnik fizički posjeduje kao što su kriptografski ključ, mobilna aplikacija, pametna kartica, token te mnogi drugi. koji zahtijevaju od korisnika izravan kontakt u trenutku verifikacije; faktor konteksta koji je definiran lokacijom korisnika, veza preko koje je korisnik spojen na mrežu identificirana je kao sigurna; faktor inherentnosti predstavljaju fizički čimbenici korisnika kao što su otisak prsta, identifikacija lica, šarenica oka, glas itd. koje je teško simulirati [4]. Nadalje, postoje mnogi tipovi metoda kao što su: autentikacija pomoću lozinke, autentikacija pomoću token-a, autentikacija pomoću pristupa treće strane (engl. *third party*), OpenID i biometrijska autentikacija. [5].

2.3 Web aplikacije

Sve opsežnijom uporabom i razvojem interneta obične web stranice transformirale su se iz jednostavnih statičnih dokumenata u kompleksnije dinamične stranice nazvane web aplikacije. Web aplikacija je računalni program kojeg pokreće korisnik, ali se ne izvršava na korisničkom računalu već na udaljenom poslužitelju; korisnik upravlja s programom preko web preglednika što je revolucionarno jer aplikacija ne zauzima memoriju rijekom rada na računalu i dostupna je sa gotovo bilo kojeg računala te omogućuje rad više korisnika u isto vrijeme [6]. Sve što je korisniku potrebno su web preglednik i mrežna veza kako bi web aplikacija mogla komunicirati sa aplikacijskim poslužiteljem. Web aplikacije široko su rasprostranjene: platforme društvenih medija koje omogućuju komunikaciju korisnicima kao što su Facebook, X, Instagram te mnoge druge; web stranice e-trgovina koje služe kao platforme za kupnju i prodaju na primjer eBay i Amazon; platforme za upravljanje sadržajem koje omogućuju korisnicima dijeljenje i upravljanje digitalnim sadržajem kao što su WordPress ili Joomle; Sustavi internetskog bankarstva koji omogućuju korisnicima lako upravljanje svojim financijama i bankovnim računima [7].

U ovome će se radu opisati najčešći oblici autentikacije, fokus će biti na autentikaciji u web aplikacijama kao što su dvofaktorska autentikacija i jednofaktorska autentikacija, autentikacija preko društvenih mreža, za primjer korištene su Google i GitHub autentikacije, te mogućnost prijave preko novčanika kripto valute Ethereum. Za implementaciju navedenih autentikacija napravljeno je jednostavno web sučelje koje će oponašati današnje društvene mreže i web aplikacije.

3. AUTENTIKACIJA U WEB APLIKACIJAMA

Velikom rasprostranjenošću web aplikacija pojavila se potreba za sigurnošću i očuvanju korisničkih podataka što je dovelo do uvođenja autentikacije u sustave web aplikacija. Autentikacija u web aplikacijama ostvaruje se prijašnje spomenutim metodama.

3.1 Jednostavna autentikacija

Autentikacija ugrađena u HTTP protokol, najosnovniji je oblik autenticiranja. Korisnička imena ili nadimci nisu šifrirana već spojeni pomoću simbola dvotočke kako bi se dobio jedan niz znakova (engl. *string*) te se taj niz kodira korištenjem baze 64. Metoda nije pouzdana jer pruža nisku razinu sigurnosti zbog nedostatka enkripcije.

```
import base64
auth = "korisnicko_ime:lozinka"
auth_bytes = auth.encode('ascii')
auth_bytes
b'korisnicko_ime:lozinka'
encoded = base64.b64encode(auth_bytes)
encoded
b'dXNlcm5hbWU6cGFzc3dvcmQ='
base64.b64decode(encoded)
b'korisnicko_ime:lozinka'
```

Slika 3.1 primjer HTTP autentikacije

3.2 Autentikacija na temelju kolačića

Autentikacija na temelju kolačića je mehanizam koji olakšava proces autentikacije i pruža ugodnije iskustvo korištenja web aplikacija. Nakon što se korisnik uspješno prijavi, poslužitelj stvori identifikaciju sesije te ga šalje klijentu kao *kolačić* odnosno malen komad teksta kojeg klijent pohranjuje i kada se korisnik kasnije želi opet autenticirati ne mora upisivati svoje faktore potvrde već poslužitelj prepozna kolačić klijenta preko kojeg je zapamtio korisnikove podatke te autenticira korisnika [8].

3.3 Autentikacija na temelju tokena

Ova metoda slična je metodi autentikacije s kolačićima međutim koriste se tokeni umjesto kolačića. Korisnik se autenticira koristeći jednostavne faktore autentikacije kao što su korisničko

ime i lozinka nakon čega mu poslužitelj vraća potpisani token koji korisnik ubuduće može koristiti kako ne bi morao ponovo upisivati svoje faktore autentikacije. JSON (engl. *JavaScript Object Notation*) Web Token je najčešće korišteni token, sastoji se od tri dijela: zaglavlje u kojem se nalazi vrsta tokena te koji je algoritam raspršivanja korišten (engl. *hashing algorithm*); korisni teret (engl. *payload*) koji sadrži tvrdnje o subjektu; potpis koji predstavlja potvrdu da poruka na putu nije bila promijenjena; sva tri su base64 kodirana i raspršena (engl. *hashed*), a token je autenticiran korištenjem metode potpisivanja privatnim ključem (engl. *private key Signature*) [9]. Token ne mora biti spremljen na poslužitelju već pri autentikaciji se potvrđuje potpisom, siguran je i kompaktan način autentikacije koji se sve više pojavljuje u praksi.

```
ZAGLAVLJE: ALGORITAM I TIP TOKENA
{
"alg": "HS256", "typ": "JWT"
}
KORISNI PODACI
}
"sub": "1234567890" "ime": "Jakob Trelec", "iat": 1516239022
POTVRDA POTPISA
HMACSHA256(
base64UrlEncode (zaglavlje) + base64UrlEncode(KORISNI PODACI), your-256-bit-secret
) secret base64 encoded
```

Slika 3.2 prikaz dekodiranog JSON tokena

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.ey
JzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6Ikpva
G4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.SflKx
wRJSMeKKF2QT4fwpMeJf36POk6yJV_adQssw5c
```

Slika 3.3 prikaz kodiranog JSON tokena

3.4 Autentikacija pristupom treće strane

Društvene mreže i mnogi davatelji usluga na internetu kao što su Facebook, Google, GitHub i slični, osim osnovnih usluga, nude i svoja aplikacijska programska sučelja (engl. *Application Programming Interface, API*) koja programer može iskoristiti kako bi dodao metodu autentikacije u svoju web aplikaciju od čega je nastala metoda pristupa treće strane. Pristup treće strane koristi protokol otvorene autentikacije (engl. *Open Authentication, OAuth*) koji omogućuje laku prijavu preko društvenih mreža u web aplikacijama. Naime, OAuth protokol

koristi se za autorizaciju korisnika, a ne za autentikaciju, razlika je što je autentikacija proces provjere identiteta korisnika, a autorizacija određuje koji korisnik smije pristupiti kojim resursima. Proces se odvija na poslužitelju društvene mreže ili davatelja usluga nakon koje korisnik daje dopuštenje davatelju usluge da podijeli identifikacijske podatke sa drugom web aplikacijom na koju će se kasnije prijavljivati, nakon autenticiranja, sljedeći put kada se korisnik bude htio prijaviti u neku web aplikaciju na kojoj je omogućen pristup treće strane, korisnik odabire preko kojeg od pružatelja treće strane će se prijaviti te ga se preusmjerava na tog pružatelja, korisnik prihvaća dijeljenje identifikacije te je preusmjeren natrag gdje se izdaje token s kojim davatelj usluge, na koju se korisnik želi prijaviti, može ubuduće identificirati korisnika [10]. Pristup treće strane omogućuje programerima laku implementaciju autentikacije, a korisnicima laku, brzu i bezbrižnu prijavu u razne sustave.

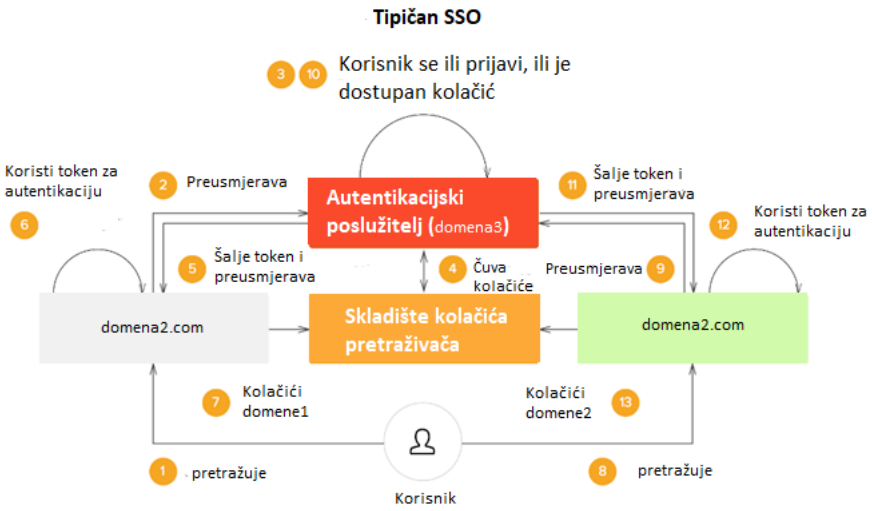
3.5 SSO autentikacija

Često se dogodi da isti tim za razvoj web aplikacija napravi više web aplikacija s različitim domenama, to dovodi do problema autentikacije korisnika koji se za istog davatelja usluga mora ponovo prijavljivati na svaku različitu domenu koju davatelj usluga pruža. SSO(engl. *Single Sign On*) rješava problem kada je potrebno omogućiti da je korisnik prijavljen na više domena i koristi iste podatke za prijavu, odnosno kada se autenticira na jednoj domeni automatski je i na drugoj. Podaci su spremljeni u informacijama sesije te je glavni zadatak SSO-a nekako podijeliti informacije sesije između domena. Način na koji se dijele informacije sesije razlikuje se s obzirom na vrstu SSO protokola, ali u suštini koncept dijeljenja ostaje isti; postoji središnja domena na koju se preusmjerava korisnike sa sporednih domena, na središnjoj domeni korisnici se autenticiraju te su preusmjereni natrag na sporednu domenu na kojoj su prvotno bili, a zatim se informacije sesije podijele na sporedne domene tako da kada im korisnik pristupi automatski je prijavljen [11]. Načini dijeljenja informacija sesija su: već spomenutim JSON Web Token-om, kolačićima sesija, SAML protokolom (engl. *Security Assertion Markup Language*), OAuth protokolom.

3.6 OTP autentikacija

OTP (engl. *One Time Password*) je nasumično generiran skup slova ili brojeva koji je stvoren kako bi omogućio korisnicima jednokratni pristup u sustav, u usporedbi s običnim lozinkama OTP nisu istog oblika već se svaki put mijenjaju kada se korisnik pokuša prijaviti. Nakon što se generira OTP šalje se korisniku hardverskim ključem, SMS-om, e-poštom ili nekog drugog

servisa za razmjenu poruka. Postoje dvije vrste OTP-ova: TOTP (engl. *Time-based one-time password*) koji se temelji na vremenu, generirani skup mijenja se nakon određenog vremenskog razdoblja, najčešće 30 sekundi; HOTP (engl. *HMAC-based one-time password*), gdje slovo H označava funkciju hash, je vrsta OTP-a u kojem se izmjena generiranog skupa znakova temelji na faktoru brojača koji se povećava svaki put kada se zatraži novi OTP, brojač je pohranjen na poslužitelju i na tokenu u kojem je generirani skup znakova, brojač na poslužitelju se povećava svaki put kada je OTP uspješno potvrđen, a na tokenu svaki put kad je zatražen; TOTP je općenito sigurniji od HOTP-a jer je otporniji na napade silom (engl. *brute force*)[12]. Često se OTP autentikacija koristi kao jedan od potvrdnih faktora u višefaktorskoj autentikaciji.



Slika 3.4 SSO autentikacija

3.7 Biometrijska autentikacija

Biometrijska autentikacija je metoda koja identificira korisnika tako što uspoređuje jedinstvene fizičke faktore korisnika ili njegove karakteristike ponašanja. Jedna je od najpouzdanijih metoda jer omogućuje brzo i precizno identificiranje korisnika na temelju njihovih fizičkih karakteristika koje su jedinstvene za svakog pojedinca što je sigurnija alternativa u usporedbi sa korisničkim imenima, lozinkama i pinovima. Prvi korak biometrijske autentikacije je dokumentiranje podataka o određenoj biometrijskoj osobini, najčešće korištene biometrijske osobine su različiti dijelovi ljudskog tijela kao što su šarenica oka, glas, hod, otisak prsta, dlan i mnogi drugi. Nadalje, sustav pohranjuje podatke o biometrijskoj osobini u bazu podataka, kada se korisnik bude htio autenticirati sljedeći put sustav traži novi uzorak biometrijske osobine te uspoređuje

novi uzorak sa pohranjenim uzorkom u bazi podataka i ako se podudaraju auteniticira korisnika [13].

3.8 Web3 autentikacija

3.8.1 Što je web3 autentikacija

Web3 autentikacija je metoda koja se temelji na tehnologiji ulančanih blokova(engl. *Blockchain*) koja omogućuje korisnicima privatnu i jednostavnu prijavu u sustave koristeći svoj kripto novčanik. Funkcionira na način da omogući korisniku potpisivanje poruke za prijavu u sustav svojim privatnim ključem te se tako potvrđuje korisnikov identitet na *blockchainu* [14]. Ovom metodom korisnici imaju veću privatnost i kontrolu nad svojim podacima.

3.8.2 Web faze

Web3 predstavlja jednu od faza web razvoja, uz web3 postojale su faze web1 i web2. Web1 predstavlja prvu početnu fazu weba, u toj fazi web stranice nisu bile kompleksne već statične te su služile samo za prikazivanje sadržaja. Daljnjim razvojem te održavanjem Web 2.0 konferencije, web je prešao u svoju drugu fazu, u ovoj fazi stranice više nisu bile statične već su osmišljene tako da budu interaktivne i promjenjive korisničkim utjecajem. Web 3 faza omogućuje nadogradnju weba u bazu podataka s integracijom DLT(engl. *Distributed Ledger Technology*) tehnologije, korištene u *blockchain* tehnologiji. Naime, ideja je da se prebaci fokus sa prednjeg sučelja(engl. *front-end*) na pozadinu(engl. *back-end*), cilj je promijeniti način korištenja i interakcije web-a da informacije nisu u vlasništvu određene osobe već su podijeljene i prikazane na logičniji način. Značajke treće web faze su semantički web koji unaprjeđuje tehnologije za stvaranje, dijeljenje i povezivanje sadržaja putem pretraživanja i analize na temelju sposobnosti razumijevanja značenja riječi, a ne na ključnim riječima ili brojevima, uz to kombinira se umjetna inteligencija kako bi računala mogla razlikovati informacije poput ljudi i pružala brže i relevantnije rezultate. Nadalje, značajke treće faze weba su integracija weba u 3D grafike, povezanost, sveprisutnost odnosno mogućnost pristupa sadržaju sa više aplikacija te se usluge mogu koristiti posvuda [15].

4. PRAKTIČNI DIO

4.1 Picstream

U praktičnom dijelu rada napravljena je jednostavna web aplikacija koja simulira društvenu mrežu, nazvana Picstream, u kojoj će se implementirati neke od navedenih metoda: jednostavna jednofaktorska autentikacija preko korisničkog imena i lozinke, dvofaktorska autentikacija korisničkim imenom, lozinkom i potvrdnim kodom, autentikacija preko društvenih mreža Google i GitHub te autentikacija preko API-a za Ethereum kripto valutu odnosno pristupom treće strane.

4.2 Korištene tehnologije

4.2.1 Python

Python je programski jezik koji je interpretiran, objektno orijentiran i interaktivan. Uključuje module, iznimke, dinamičko tipkanje, dinamičke tipove podataka vrlo visoke razine i klase. Podržava više programskih paradigmi izvan objektno orijentiranog programiranja, kao što je proceduralno i funkcionalno programiranje. Python je jezik vrlo lako razumljive sintakse, sadrži sučelja za mnoge sistemske pozive, uz velik broj biblioteka, i za razne prozorske sustave odnosno grafička korisnička sučelja (engl. *Graphical User Interface, GUI*), proširiv je s drugim programskim jezicima te je lako prenosiv odnosno funkcionira na raznim operacijskim sustavima kao što su Windows, MacOS i mnoge distribucije Linux-a. Izmislio ga je Guido Van Rossum(rođen 1956., nizozemski programer), u Nizozemskoj, kada je 1980-ih počeo raditi na njemu kako bi naslijedio tadašnji ABC jezik [16].

4.2.2 Django

Django je besplatan Python-ov web okvir (engl. *web framework*) otvorenog koda, visoke razine koji potiče brz i čist razvoj web aplikacija sa pragmatičnim dizajnom. Izgrađen je s ciljem omogućavanja lakšeg razvoja web aplikacija tako da se programeri mogu usredotočiti samo na pisanje aplikacije gdje im je temelj aplikacije već postavljen. Django prati arhitekturu MVT (engl. *Model View Template*) koja razdvaja aplikaciju na tri dijela: Model je dio aplikacije koji je zadužen za strukturiranje i manipuliranje podacima; Pogledi (engl. *Views*) zasluženi su za rukovanje logikom procesiranja korisničkih zahtjeva te vraćanje odgovora; Šablone (engl *Templates*) odgovorne su za način na koji se informacije prikazuju korisnicima [17]. Najpopularniji je zbog svog automatski stvorenog sučelja za administratore i rukovanje autentikacijom te svojoj visokoj skalabilnosti. Nastao je 2003. kada su web programeri Lawrence

Journal-World-a, Adrian Holovaty(rođen 1981., američki programer) i Simon Willison(rođen 1981., britanski programer) počeli koristiti Python za izradu aplikacija.

models.py

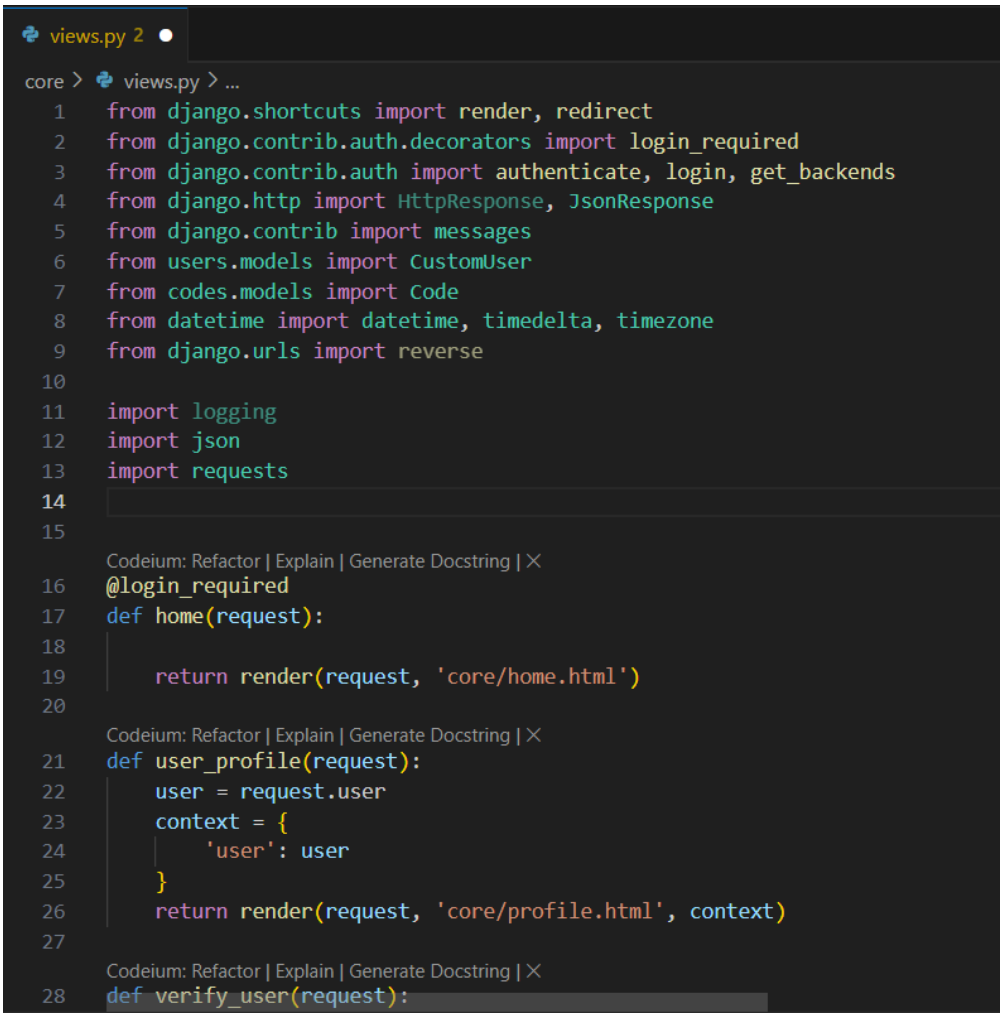
```python
from django.db import models
from django.contrib.auth.models import AbstractBaseUser


class CustomUser(AbstractBaseUser):
    email = models.EmailField(unique=True)
    zip_code = models.CharField(max_length=6)
    name = models.CharField(max_length=30)
    is_staff = models.BooleanField(default=False)
    REQUIRED_FIELDS = ['zip_code']
    USERNAME_FIELD = 'email'

    def get_short_name(self):
        return self.email

    def __str__(self):
        return self.email
```

Slika 4.1 prikaz Django modela



Slika 4.2 prikaz Django pogleda

4.2.3 HTML

HTML (engl. *Hypertext Markup Language)* je standardni označni jezik izmišljen za stvaranje web stranica. Pomoću različitih elemenata i načina označivanja omogućuje definiranje izgleda i sadržaja web stranica. HTML nije programski jezik već označni jer ne dopušta korištenje varijabli i uvjetnih izjava (engl. *Conditional Statements*). Sastoji se od elemenata koji služe za modificiranje sadržaja na web stranici te je moguće omotati sadržaj, promijeniti mu boju, font ili ga učiniti hipervezom uz još mnogo različitih mogućnosti. HTML element sastoji se od; početne oznake (engl. *Tag*) koji u sebi ima naziv elementa i označava početak elementa, uz naziv možemo u početnu oznaku stavljati razne atribute s kojim pobliže označavamo element ili ga identificiramo imenom ili klasom; sadržaja elementa koji je u većini slučajeva tekst, ali može biti

i prazan za posebne elemente; Završne oznake koja je ista kao početna samo što sadrži dodatnu kosu crtu ispred naziva elementa [18].

templates > core > profile.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>{{ user.username }}'s Profile</title>

    <link href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css" rel="stylesheet">
</head>
<body>
<div class="container mt-5">
    <div class="card">
        <div class="card-header">
            <h3>Profil: {{ user.username }}</h3>
            <a href="{% url 'home' %}" class="btn btn-secondary btn-sm">Povratak na početak</a>
        </div>
        <div class="card-body">
            <p><strong>Korisničko ime:</strong> {{ user.username }}</p>
            <p><strong>E-pošta:</strong> {{ user.email }}</p>
            <p><strong>Ime:</strong> {{ user.first_name }}</p>
            <p><strong>Prezime:</strong> {{ user.last_name }}</p>
            <p><strong>Broj mobitela:</strong> {{ user.phone_number }}</p>
        </div>
    </div>
```

Slika 4.3 Django template u HTML-u

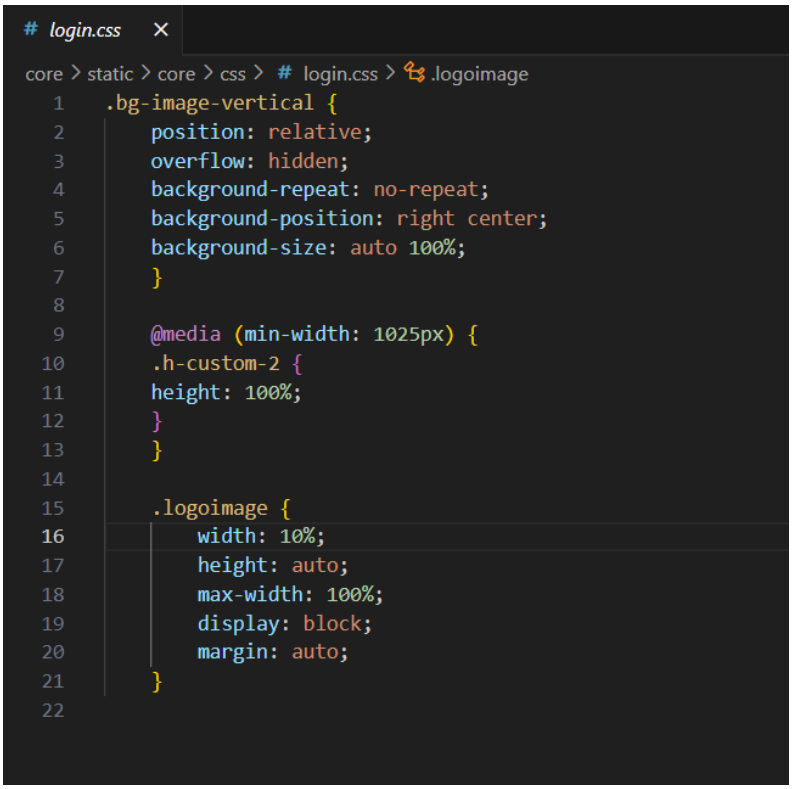
Struktura HTML dokumenta sadrži tri dijela: glava, tijelo i posebne deklaracije *DOCTYPE* . Element glave služi kao spremnik za meta podatke odnosno podatke o podacima, u njemu se opisuju fontovi, naslov dokumenta, set znakova koji će se koristiti, stilovi, skripte i tako dalje; element tijela sadrži ostale elemente s kojima ćemo opisivati sadržaj web stranice; deklaracija *DOCTYPE* obavezna je te se nalazi na vrhu HTML dokumenta, Njegova jedina svrha je spriječiti preglednik da se prebaci u takozvani *quirks mod* prilikom renderiranja dokumenta; to jest, doctype osigurava da preglednik čini najbolji pokušaj da slijedi relevantne specifikacije, umjesto da koristi drugačiji način prikazivanja koji nije kompatibilan s nekim specifikacijama.

```
<!DOCTYPE html>
<html>
<head>
    <title>Naslov stranice</title>
</head>
<body>
    <h2>Sadržaj naslova</h2>
    <p>Sadržaj odlomka</p>
</body>
</html>
```

Slika 4.4 raspored HTML dokumenta

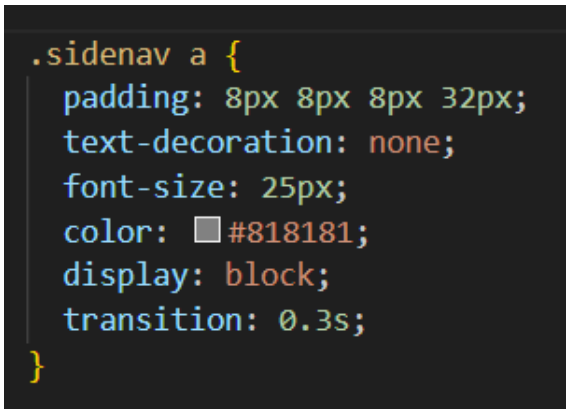
4.2.4 CSS

CSS (engl. *Cascading Style Sheets*) jezik je deklarativnog stila koji se koristi za dizajn sadržaja na web stranicama. HTML dodaje sadržaj na web stranice te je u mogućnosti blago ga dizajnirati dok CSS omogućuje dodatno stiliziranje prikaza web sadržaja. CSS prvi se put pojavljuje 1994. kada ga je predložio Norvežanin Håkon Wium Lie(rođen 1965., norveški web dizajner i aktivist) te dvije godine nakon *World Wide Web Consortium* (engl. *W3C*) usvaja prve standardizirane specifikacije nazvane CSS1 koju su razvili Håkon Wium Lie i Bert Bos(rođen 1963., nizozemski računalni znanstvenik). Microsoftov Internet Explorer je prvi komercijalni preglednik koji podržava CSS. Godine 1998. objavljen je CSS2, koji nudi poboljšanu kontrolu izgleda i mogućnost određivanja kako će se sadržaj pojaviti na određenim platformama kao što su ručni uređaji zaslon, televiziju i Brailleovo pismo. CSS3 je uveden 2011. te su dodane funkcije kao što je responzivan web dizajn i podrška za mnoge nove vrste fontova [19]. CSS je jedan od temeljnih jezika otvorenog weba i standardiziran je za sve web preglednike prema W3C specifikacijama.



Slika 4.5 CSS

CSS kod funkcionira tako da koristi selektore preko kojih identificira HTML elemente na koje će se promjene primijeniti, nakon selektora idu vitičaste zagrade u kojima se definiraju deklaracije koje predstavljaju atribute koji će biti primijenjeni na HTML element, deklaracije se sastoje od svojstava i vrijednosti. Za CSS se kaže da je kaskadan jer se stilska pravila u CSS kodu nižu od jednog do drugog u hijerarhijskom poretku, u kojem stilska pravila kodirana na višem položaju imaju prednost nad pravilima niže u kodu.



Slika 4.6 sintaksa CSS-a

4.2.5 Bootstrap

Bootstrap je besplatni alat otvorenog koda za razvoj web aplikacija koji Sadrži HTML, CSS i predloške dizajna temeljene na JavaScriptu za tipografiju, obrasce, gumbe, navigaciju i druge komponente korisničkih sučelja. Bootstrap su razvili Mark Otto(rođen 1986., američki web dizajner i programer) i Jacob Thornton(američki web dizajner i programer) na Twitter-u, a objavljen je kao projekt otvorenog koda u kolovozu 2011. Izvorno poznat kao Twitter Bootstrap, imao je za cilj standardizirati razvojni proces u Twitter-ovim internim alatima [20]. Boostrap je popularan među web programerima jer omogućuje veliku zbirku koda te brz i jednostavan pronalazak dizajna za HTML komponente koji mogu samo ubaciti u svoj kod i uštediti vrijeme. Boostrap funkcionira tako da samo dodamo HTML element za hipervezu u glavu HTML dokumenta koristeći CDN(engl. *Content Delivery Network*) koji stavljamo tu *href* atribut elementa *link*.

```
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
    <link rel="stylesheet" href="{% static 'core/base.css' %}">

    <title>Picstream</title>
```

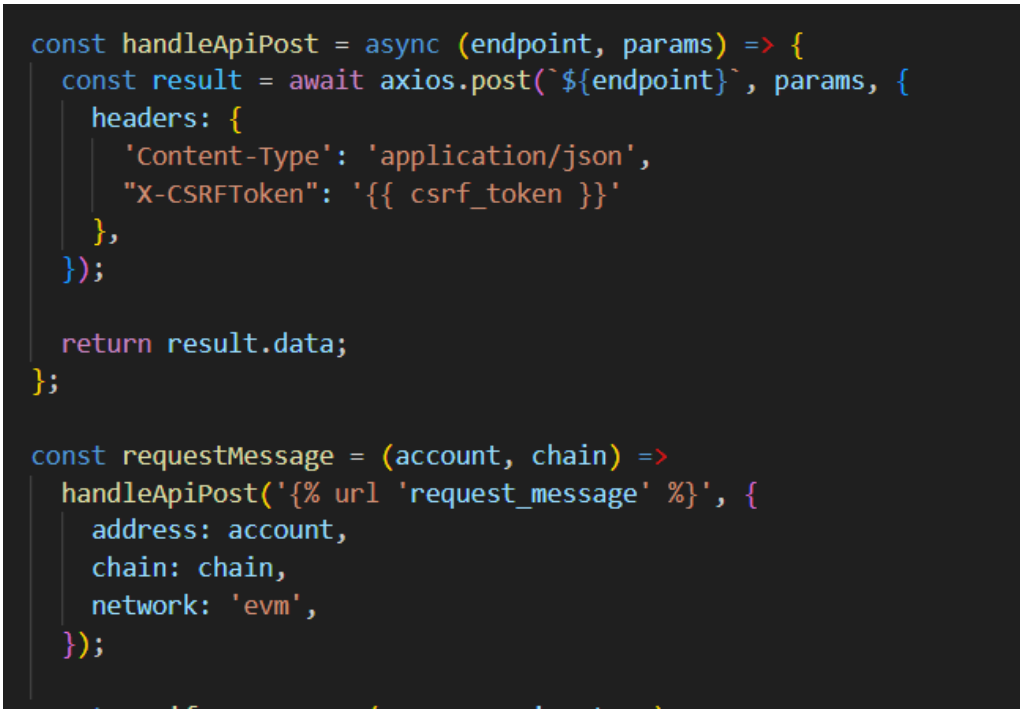
Slika 4.7 dodavanje Boostrapa

Alternativan način za dodavanje Bootstrap-a u kod je preuzimanjem datoteka sa službene stranice i uključivanjem u direktorij projekta. JavaScript ima bogatu povijest koja odražava njegovu evoluciju od jednostavnog skriptnog jezika do moćnog programskog jezika koji se široko koristi na webu. .

4.2.6 JavaScript

JavaScript je interpretirani programski jezik koji programerima omogućuje stvaranje dinamičkih i interaktivnih web sučelja uz HTML i CSS te je jedan od temeljnih tehnologija WWW-a (engl. *World Wide Web*). Marc Andreessen(rođen 1971., američki razvijač softvera), osnivač Netscape Communication-a, imao je viziju da je webu potreban način da postane dinamičniji, zamislio je da će animacije, interaktivnost te automatizacija biti budućnost web dizajna; na pamet mu je pao jednostavan skriptni jezik koji je ciljano zamišljen za dizajnere kojima nije potrebna velika količina znanja i iskustva u programiranju, tako je nastao Mocha, dostupan jednostavan skriptni jezik za web; Brendan Eich(rođen 1961., američki programer), otac JavaScript-a, koji je isto

radio za Netscape Communication u razvoju sheme za web preglednik odlučio je udružiti snage. Prototip Mocha nakon svoje integracije u Netscape Communicator 1995. Preimenovan je u LiveScript, a kasnije iste godine Netscape Communications i Sun Microsystems, autor programskog jezika Java, sklopili su ugovor u kojem je dogovoreno da će se LiveScript preimenovati u JavaScript te bi bio predstavljen kao skriptni jezik za male klijentske zadatke u pregledniku, dok će Java biti profesionalni alat za veće zadatke i razvoj bogatih web komponenti. [21].



Slika 4.8 primjer JavaScript koda

JavaScript je dinamičan, interpretiran programski jezik koji je u svojim počecima prepoznat kao skriptni web jezik ubrzo se počeo koristiti i u drugim okruženjima i tehnologijama kao što su Node.js , Adobe Acrobat, QuickJS i ostali.

4.2.6 Moralis programsko sučelje za autentikaciju

Moralis programsko sučelje (engl. *Application Programming Interface, API*) je dio Moralis platforme koja pruža korisnicima autentikaciju i provjeru potpisanih poruka sa svojim Web3 novčanicima. Moralis platforma razvija alate za integraciju funkcionalnosti temeljenih na *blockchain-u* kao što su autentikacija, pohrana podataka i interakcija pametnog ugovora u

decentralizirane aplikacije. Na slici 4.9. prikazan je način integracije Moralis programskog sučelja u web aplikaciju.

```
def request_message(request):
    data = json.loads(request.body)
    print(data)

    present = datetime.now(timezone.utc)
    present_plus_one_m = present + timedelta(minutes=1)
    expirationTime = str(present_plus_one_m.isoformat())
    expirationTime = str(expirationTime[:-6]) + 'Z'

    REQUEST_URL = 'https://authapi.moralis.io/challenge/request/evm'
    request_object = {
      "domain": "127.0.0.1",
      "chainId": 1,
      "address": data['address'],
      "statement": "Please confirm",
      "uri": "http://127.0.0.1:1000/",
      "expirationTime": expirationTime,
      "notBefore": "2020-01-01T00:00:00.000Z",
      "timeout": 15
    }
    x = requests.post(
        REQUEST_URL,
        json=request_object,
        headers={'X-API-KEY': API_KEY})

    return JsonResponse(json.loads(x.text))
```

Slika 4.9 integracija Moralis sučelja

4.3 Implementacija

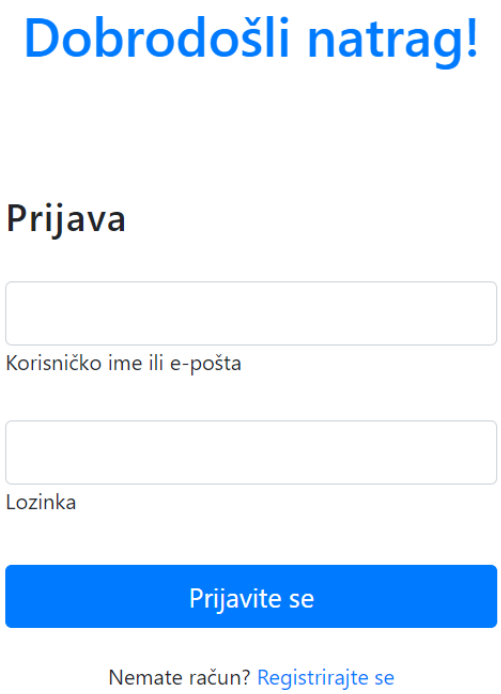
Za pokretanje Web aplikacije koristi se lokalni *host* te se na nju pristupa preko hiperveze: http://127.0.0.1:1000/ . Kada korisnik uđe na web mjesto automatski je preusmjeren na početnu stranicu, ali za za pristup početnoj stranici i ostalom sadržaju korisnik mora biti prijavljen, te je automatski opet preusmjeren na stranicu za prijavu.

Slika 4.10 biranje načina prijave

Na slici 4.8 vidi se izgled stranice za biranje načina prijave. U aplikaciji implementirani su primjeri najčešćih metoda prijave na web aplikacijama, kao što se može uočiti korisnik može odabrati između prijave u sustav koristeći metodu treće strane, u ovom slučaju preko Google-ovih usluga ili GitHub-a. Nadalje, korisnik može odabrati običnu metodu prijave korisničkim imenom i lozinkom te numeričkog koda koji mora upisati nakon upisivanja korisničkom imena i lozinke, dvofaktorska autentikacija. Naposljetku, dodana je posebna moderna metoda autentikacije korištenjem Ethereum kripto novčanika, web3 autentikacija.

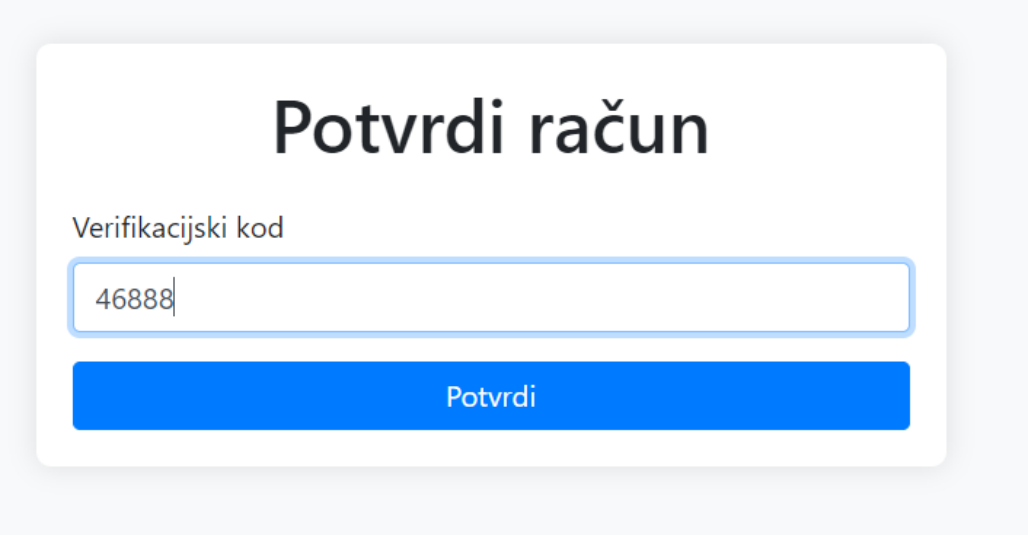
4.3.1 Prijava korisničkim imenom i lozinkom sa/bez kodom za potvrdu

Odabirom prve opcije, prema slici 4.9., korisnik bira običan način prijave korisničkim imenom i lozinkom. Ovaj način prijave visoke je učinkovitosti i brzine, korisnicima je najpoznatiji jer je se već dugo vremena koristi i većina korisnika je upoznata sa procedurom prijave, ali jednostavnost i brzina dolaze s troškom manje sigurnosti te je u usporedbi s ostalim implementiram metodama najmanje siguran.



Slika 4.11 obična prijava

Uvođenjem dvofaktorske autentikacije, numerički kod za potvrdu koji se korisniku šalje e-poštom, SMS-om ili Google autentikatorom, znatno se povećava sigurnost ove metode. Naime, u ovom slučaju napadač osim korisničkog imena i lozinke mora znati i potvrdni kod koji se šalje u trenutku prijave što znači da napadač mora imati pristup ili korisnikovom mobilnom telefonu ili njegovoj e-pošti odnosno Google uslugama.

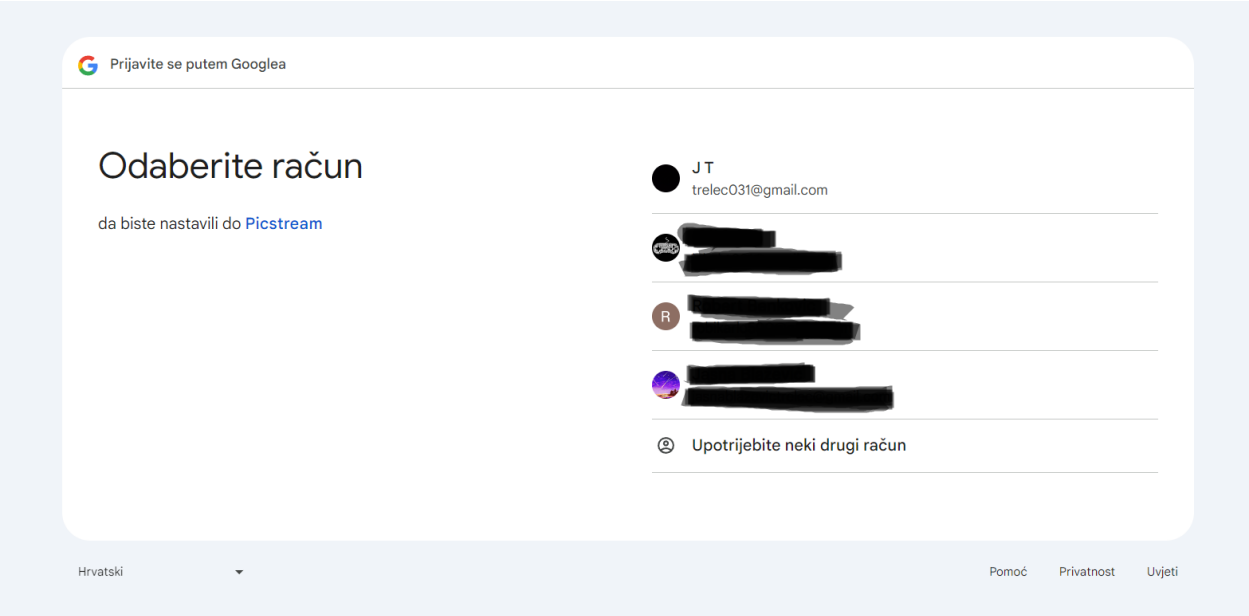


Slika 4.12 dvofaktorska potvrda

4.3.2 Prijava korištenjem Google-ovih usluga

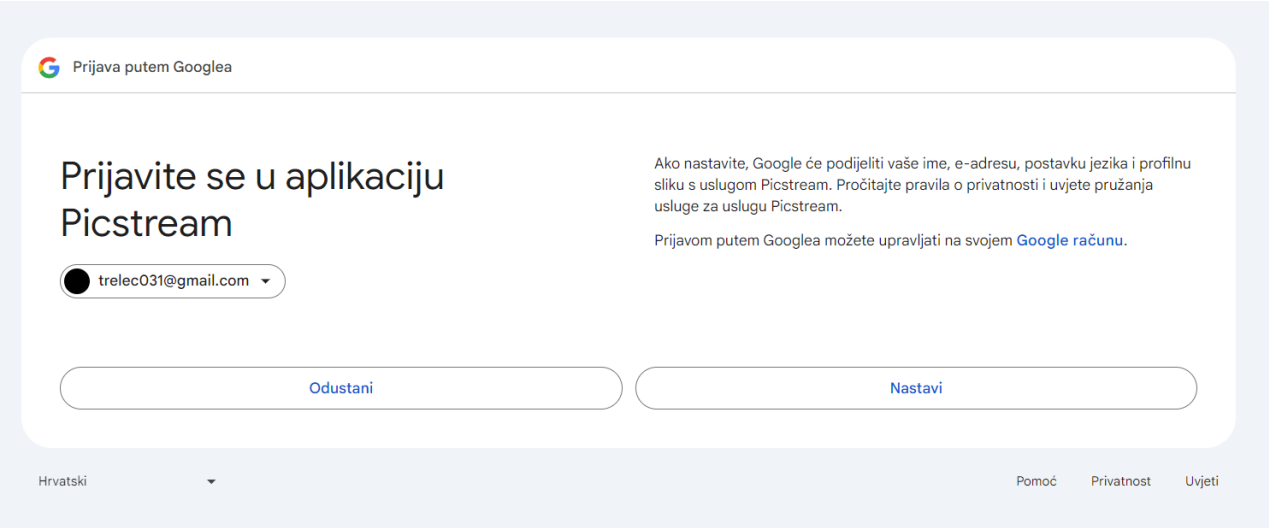
Druga opcija, prema slici 4.9., predstavlja metodu prijave preko Google-ovih usluga, pristupom treće strane. Ovo je moderna metoda koja je već široko u uporabi te je poznata korisnicima, vrlo je visoke učinkovitosti jer prijava traje samo par sekundi i u nekoliko klikova mišem, pogotovo ako je korisnik već prijavljen u Google-ove usluge.

Prijava se odvija tako da korisnik klikne na opciju prijava preko Google-ovih usluga, zatim je korisnik preusmjeren na sljedeću stranicu gdje korisnik bira s kojim računom će se prijaviti.



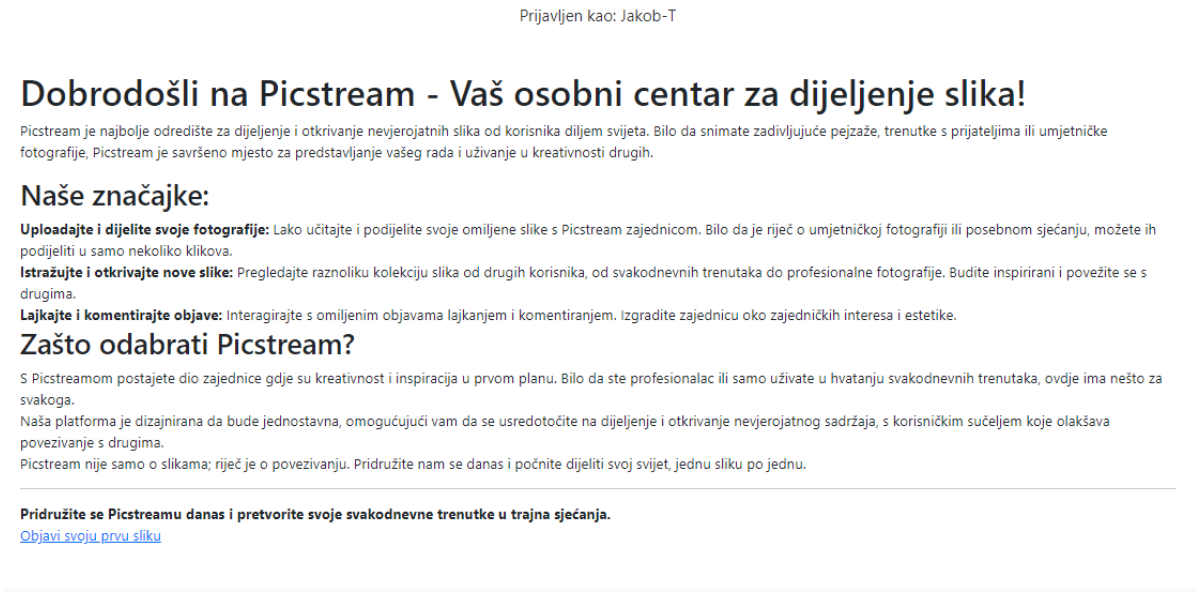
Slika 4.13 biranje profila za prijavu

Nakon odabira računa preko kojeg se korisnik želi prijaviti slijedi sljedeći prikaz u kojem korisnik potvrđuje da se želi prijaviti u aplikaciju što se može vidjeti na slici 4.12.



Slika 4.14 potvrda suglasnosti prijave

Nakon uspješne prijave korisnik je preusmjeren na početnu stranicu.

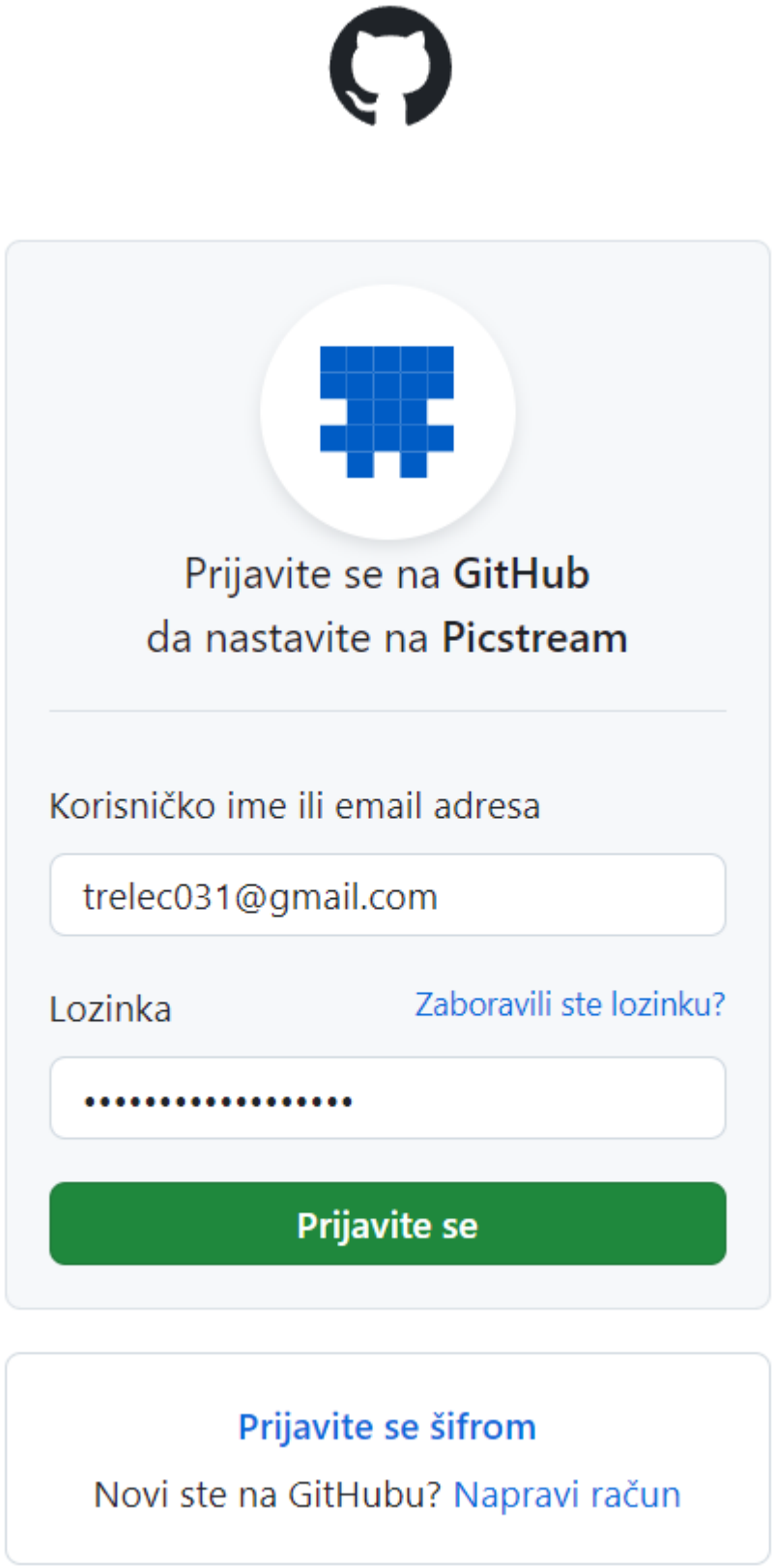


Slika 4.15 početna stranica

4.3.3 Prijava korištenjem GitHub-ovih usluga

Treća opcija, prema slici 4.9., predstavlja metodu autentikacije preko GitHub-ovih usluga. Metoda je slična kao i kod Google-ovih usluga, moderna, učinkovita, ali manje popularna što ovisi o popularnosti davatelja usluga, u ovom slučaju platforma GitHub koja je najpopularnija kod programera. Koraci prijave slični su kao i kod Google prijave.

Prvi korak je da korisnik klikne na opciju prijava preko GitHub-a, nakon toga korisnik je preusmjeren na sljedeću stranicu gdje korisnik bira s kojim računom će se prijaviti. Zatim, korisnik potvrđuje da želi odobriti prijavu svojim GitHub računom na stranici web aplikacije.

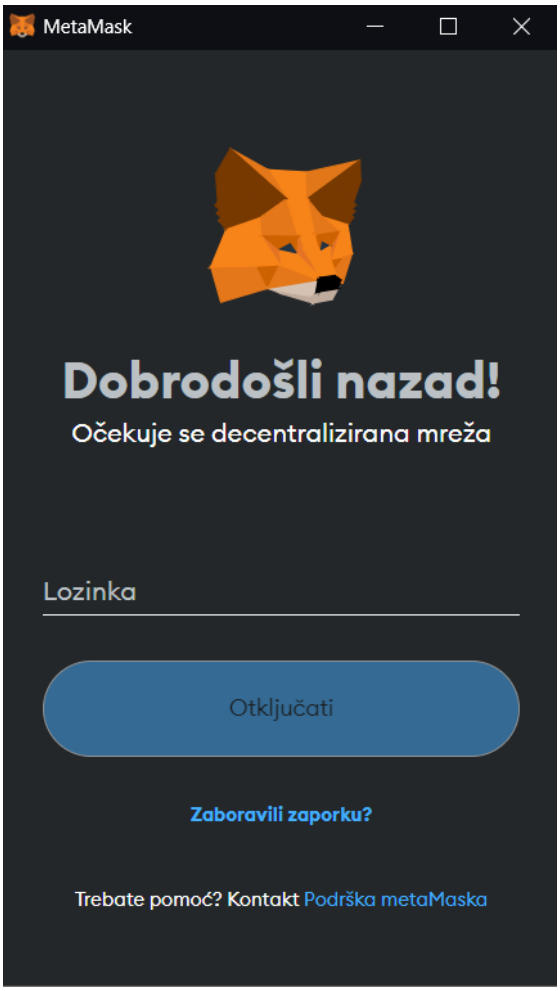


Slika 4.16 GitHub prijava

4.3.4 Prijava korištenjem kripto novčanika

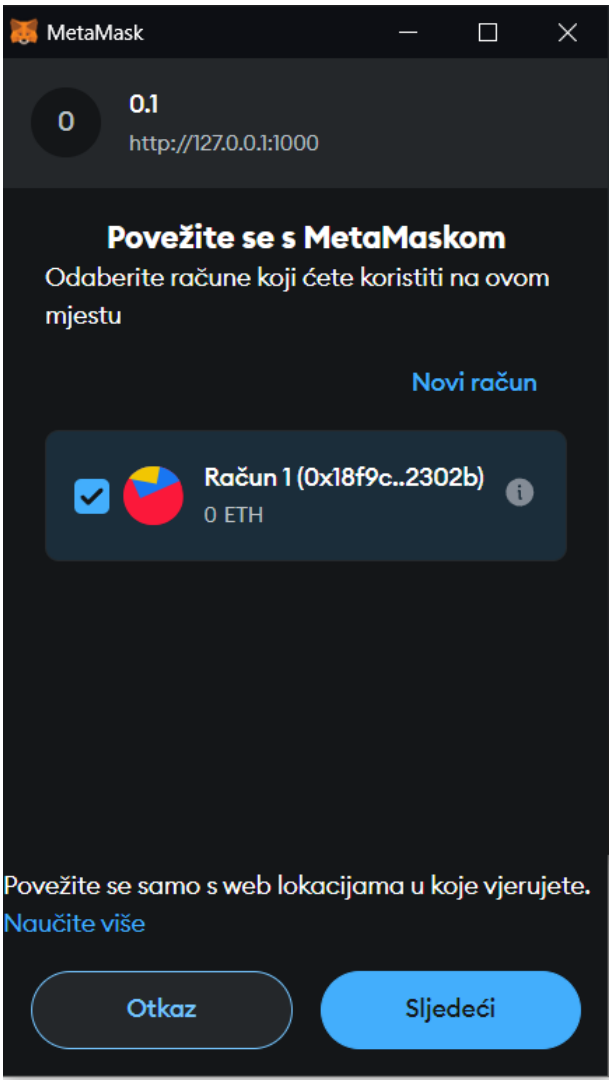
Odabirom četvrte opcije, prema slici 4.9., korisnik bira prijavu korištenjem Ethereum kripto novčanika. Za ovu metodu korisnik mora posjedovati MetaMask Ethereum kripto novčanik te MetaMask proširenje na svome web pretraživaču. Ova metoda pokazala se kao iznimno sigurna, ali učinkovitost je srednje visoka zbog kompliciranijeg pristupa autenticiranja, od korisnika se zahtjeva da posjeduje dodatno web proširenje uz kripto novčanik te proces prijave traje malo duže u usporedbi sa ostalim metodama.

Prvi korak je da korisnik odabere četvrtu opciju te će se zatim otvoriti mali prozor MetaMask proširenja u gornjem desnom kutu web pretraživača.

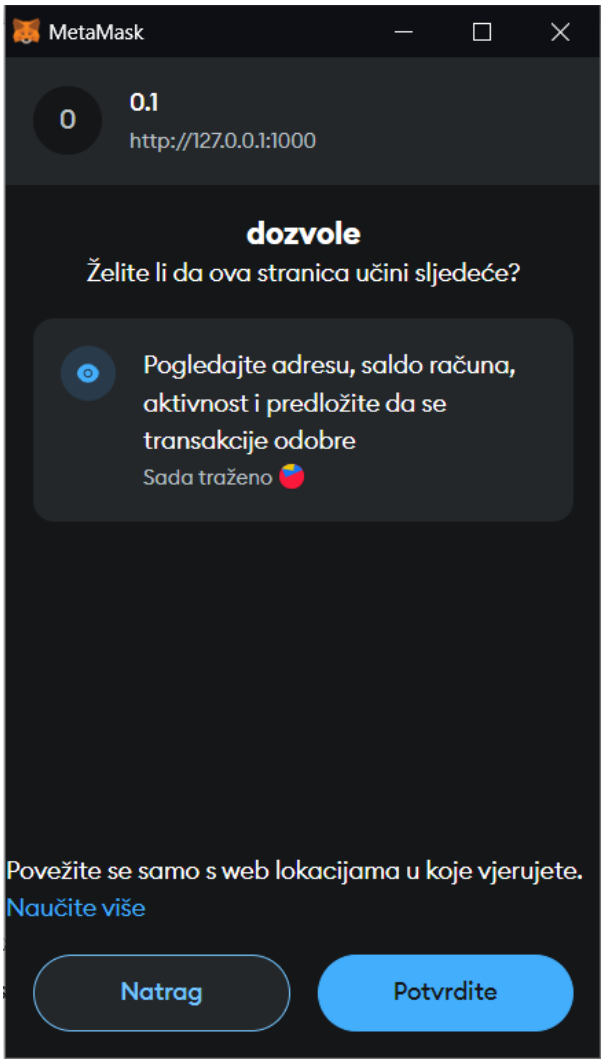


Slika 4.17 MetaMask prozor

Nakon što se otvori prozor korisnik se mora prijaviti svojim MetaMask računom na kojemu se nalaze različiti kripto novčanici ili ako je već prijavljen ovaj korak se preskače. Nadalje, korisnik bira kojim novčanikom će se prijaviti (slika 4.15.), nakon čega potvrđuje prijavu za odabrani novčanik.

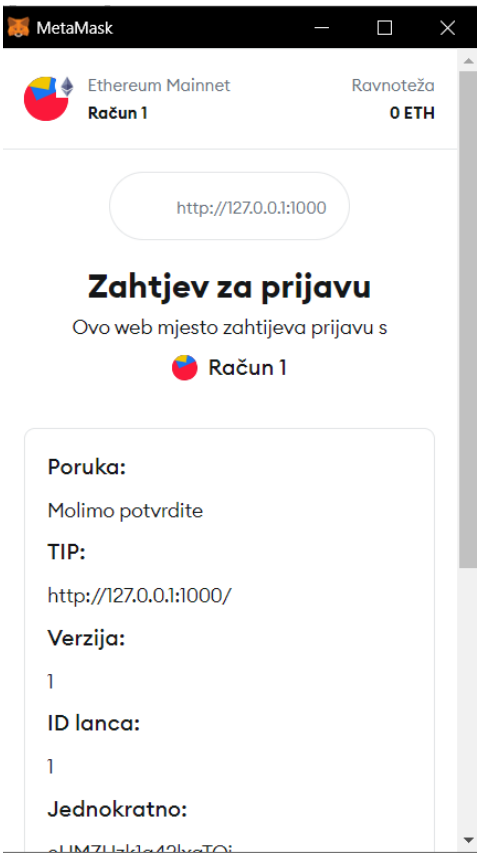


Slika 4.18 odabir kripto novčanika

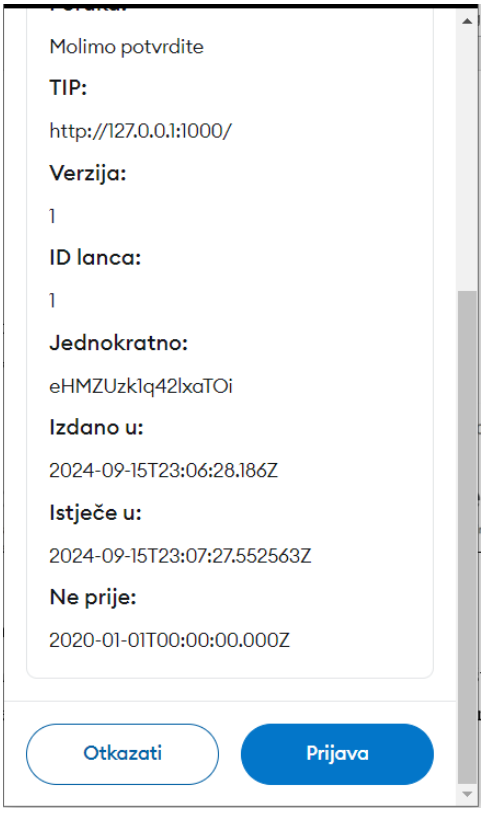


Slika 4.19 dopuštenje za prijavu

Nakon odabira novčanika na prozoru se prikažu informacije o stranici na koju se korisnik prijavljuje te detalji o sesiji, ovdje korisnik potvrđuje da se uistinu želi prijaviti na web aplikaciju.



Slika 4.20 prozor za potvrdu



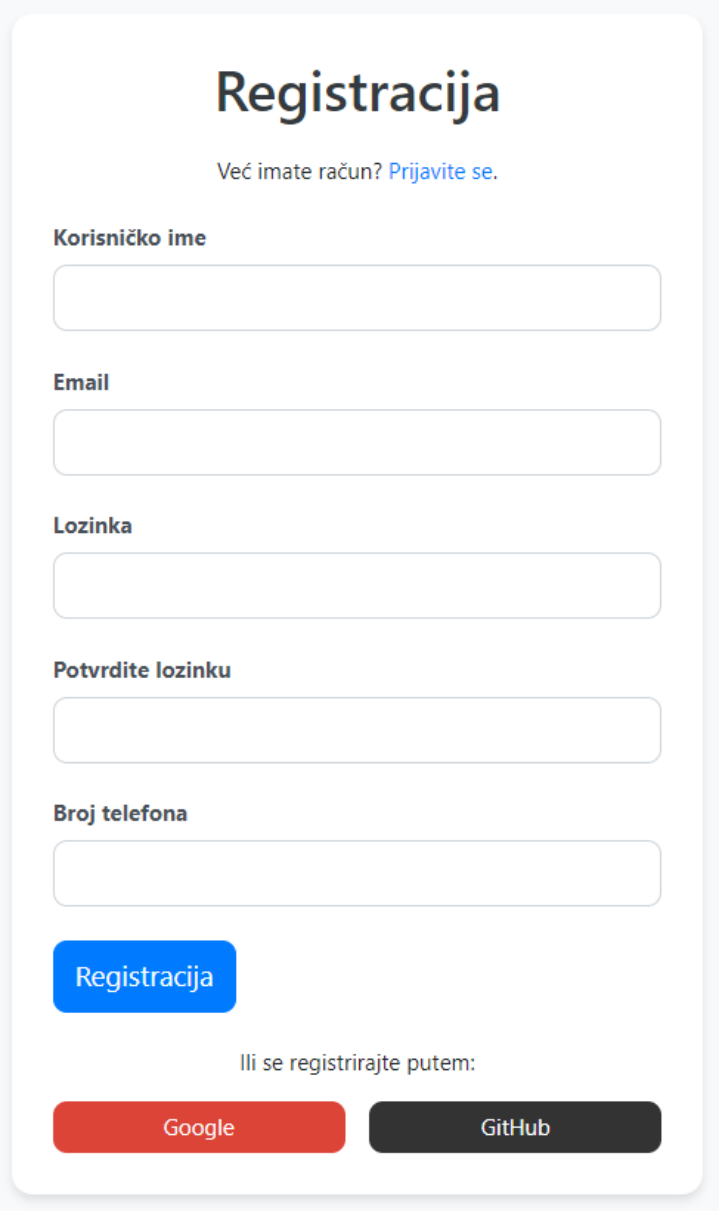
Slika 4.21 drugi prozora za potvrdu

Korisnik prijavljen kripto novčanikom ne posjeduje korisničko ime već se identificira kripto adresom što je uočljivo u gornjem dijelu slike 4.21.

Slika 4.22 prijava kripto novčanikom

4.3.5 Registracija

Za registraciju korisnika koristi se jednostavan obrazac registracije. Kao što je vidljivo na slici 4.19., korisnik upisuje korisničko ime, adresu e-pošte, lozinku, potvrdu lozinke i broj mobitela.



Slika 4.23 registracija

5. ZAKLJUČAK

U ovom završnom radu istražile su se te implementirale razne metode autentikacije u web aplikacijama kako bi se procijenio njihov rad i pouzdanost u kontekstu korisničke sigurnosti i lakoće korištenja. Analizirane su razne metode te su određene bile i implementirane, implementacijom te dodatnom analizom provjeravala se učinkovitost i sigurnost određenih metoda. Analizom tradicionalne metode prijave, korisničkim imenom i lozinkom, ustanovljeno je da su iznimno učinkovite zbog svoje lakoće korištenja i sveprisutnošću, ali slabije su sigurnosti u usporedbi sa ostalim metodama i podložne na razne napade. Prijava korisničkim imenom i lozinkom se može dodatno osigurati ukoliko se uvede dodatan faktor potvrde, kao što je numerički kod, koji znatno povećava sigurnost uz smanjenje učinkovitosti jer se uvodi dodatan korak što povećava kompleksnost prijave. Nadalje, ispitane su vrste autentikacije treće strane, korištenjem usluga platformi GitHub i Google, koje se se pokazale kao metode visoke učinkovitosti jer omogućuju korisnicima prijavu uz par klika mišem te su visoke sigurnosti jer se oslanjaju na kompleksne algoritme zaštite GitHub-a i Google-a. Zadnja ispitana metoda je koristeći kripto novčanik koja se pokazala kao metoda visoke sigurnosti koja omogućava korisnicima i dodatnu privatnost, ali zahtjeva više koraka te dodatno proširenje web pretraživača. Rezultati analize pokazali su da su prijave korištenjem treće strane i kombinacija tradicionalne metode prijave korisničkim imenom i lozinkom uz dodatan faktor prijave, kao što su kodovi ili niz znakova, najbolji omjer sigurnosti i učinkovitosti. U web aplikacijama preporučuje se implementacija više metodi što bi pružalo korisnicima jednostavnije i ugodnije iskustvo korištenja, a preporučene metode su višefaktorska i dvofaktorska autentikacija te autentikacija pristupom treće strane. Zaključno s tim, ovaj rad pridonosi razumijevanju raznih metoda autentikacije te pruža korisne informacije za implementiranje istih u web aplikacijama. Nadalje, ovaj rad može usmjeriti buduća istraživanja na analizu i razvijanje naprednih metoda autentikacije kako bi se poboljšalo korisničko iskustvo.

LITERATURA

[1] WorkOS, „A Developer's History of Authentication“ [online], WorkOS, Inc., 5-prosinca-2020. Dostupno na: https://workos.com/blog/a-developers-history-of-authentication [Pristupljeno 03.09.2023.].

[2] Andrew Magnusson, „The Definitive Guide to Authentication: History of Authentication“ [online], StrongDM, zadnje ažurirano 30-kolovoza-2024. Dostupno na: https://www.strongdm.com/authentication [Pristupljeno 3.9.2024.].

[3] Nick Barney, Mary E. Shacklett, Linda Rosencrance, „Authentication: How does authentication work?“ [online], TechTarget Inc., zadnje ažurirano studeni-2023. Dostupno na: https://www.techtarget.com/searchsecurity/definition/authentication [Pristupljeno: 3.9.2024.].

[4] NordLayer, „MFA vs. 2FA: what's the difference?“ [online], NordLayer, 20-rujna-2022. Dostupno na: https://nordlayer.com/blog/mfa-vs-2fa-whats-the-difference/ [Pristupljeno 3.9.2024.].

[5] Sandip Roy, „Popular Authentication Methods for Web Apps“ [online], Baeldung, zadnje ažurirano 18-ožujka-2024. Dostupno na: https://www.baeldung.com/cs/authentication-web-apps [Pristupljeno 3.9.2024.].

[6] Adam Volle, „Web application“ [online], Encyclopædia Britannica, Inc., Dostupno na: https://www.britannica.com/topic/Web-application [Pristupljeno 4.9.2024.].

[7] Matt Hicks, „What is a web application - what you need to know in 2023: What are some common web applications?“ [online], Lightflows, 24-rujna-2023. Dostupno na: https://www.lightflows.co.uk/blog/what-is-a-web-application-what-you-need-to-know-in-2023/ [Pristupljeno 4.9.2024.].

[8] Matea Dražin, ULOGA KOLAČIĆA U PERSONALIZACIJI I PRILAGODBI WEB

APLIKACIJA, Repozitorij Prirodoslovno-matematičkog fakulteta u Splitu, Split 2023., Dostupno na: https://repozitorij.pmfst.unist.hr/islandora/object/pmfst:1773/datastream/PDF/view [Pristupljeno 6.9.2024.].

[9] Amal Shaji, „Web Authentication Methods Compared: Token-Based Authentication“ [online], TestDriven.io, zadnje ažurirano 10-veljače-2023. Dostupno na: https://testdriven.io/blog/web-authentication-methods/ [Pristupljeno 6.9.2024.].

[10] Susan Morrow, „Third-party authentication (OAuth): Good or bad for security?: How does third-party authentication work?“ [online], InfoSec Institute, 25-srpnja-2022. Dostupno na: https://www.infosecinstitute.com/resources/industry-insights/third-party-authentication-oauth-good-or-bad-for-security/ [Pristupljeno 6.9.2024.].

[11] Sebastian Peyrott, „What Is Single Sign-On Authentication (SSO) And How Does It Work? : Single Sign-On (SSO) Authentication“ [online], Okta, Inc., zadnje ažurirano 13-ožujka-2023. Dostupno na: https://auth0.com/blog/what-is-and-how-does-single-sign-on-work/ [Pristupljeno 7.9.2024.].

[12] Descope, „OTP Authentication Explained: Definition, Uses & Benefits: Types of OTPs“ [online], Descope, 30-kolovoza-2023. Dostupno na: https://www.descope.com/learn/post/otp [Pristupljeno 7.9.2024.].

[13] Ondato, „A Complete Guide to Biometric Authentication Methods: How Does Biometric Authentication Work?“ [online], Ondato, 13-travnja-2023. Dostupno na: https://ondato.com/blog/benefits-of-biometric-authentication/ [Pristupljeno 8.9.2024.].

[14] Hitesh Sant, „10 Web3 Auth Platforms to Secure Your User Login Process [2024]: What is Web3 Authentication?“ [online], Geekflare, zadnje ažurirano 2-siječnja-2024. Dostupno na: https://geekflare.com/web3-auth-platforms/ [Pristupljeno 10.9.2024.].

[15] Nada Khaled, „The Difference Between Web 1, Web 2 , Web 3 and Web 4.“ [online], Medium, 24-rujna-2023. Dostupno na: https://nadakhaliedsaid.medium.com/the-difference-between-web-1-web-2-web-3-and-web-4-9238e4853aec [Pristupljeno 10.9.2024.].

[16] Python, „What is Python?“ [online], Python, Dostupno na: https://docs.python.org/3/faq/general.html#what-is-python [Pristupljeno 8.9.2024.].

[17] MDN Web Docs, „Django introduction: What is Django?“ [online], MDN Web Docs, Dostupno na:https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/Introduction [Pristupljeno 8.9.2024.].

[18] MDN Web Docs, „HTML basics: So what is HTML?“ [online], MDN Web Docs, Dostupno na: https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/HTML_basics [Pristupljeno 8.9.2024.].

[19] André Munro, „CSS: History“ [online], Encyclopædia Britannica, Inc., zadnje ažurirano 6-rujna-2024. Dostupno na: https://www.britannica.com/technology/CSS-programming-language [Pristupljeno 8.9.2024.].

[20] Alexandre Ouellette, „What is Bootstrap: A Beginner's Guide: What is Bootstrap?“ [online], CareerFoundry GmbH, zadnje ažurirano 2-veljače-2023. Dostupno na: https://careerfoundry.com/en/blog/web-development/what-is-bootstrap-a-beginners-guide/ [Pristupljeno 8.9.2024.].

[21] Sebastian Peyrott, „A Brief History of JavaScript“ [online], Okta, Inc., 16-siječnja-2017. Dostupno na: https://auth0.com/blog/a-brief-history-of-javascript/ [Pristupljeno 9.9.2024.].

SAŽETAK

Završni rad istražuje i implementira različite metode autentikacije u web aplikacijama kako bi se ispitala njihova učinkovitost i pouzdanost u pogledu korisničke sigurnosti i jednostavnosti korištenja. Cilj rada bio je pronaći najučinkovitiju i najsigurniju metodu autentikacije za web aplikacije. Istraživanje je uključilo analizu raznih metoda autentikacije te implementaciju najčešće korištenih metoda u jednostavno sučelje kako bi se mogle dodatno ispitati i analizirati. Implementiranjem odabranih metoda te njihovom dodatnom analizom utvrđeno je koje su metode autentikacije najbolje u kontekstu sigurnosti i lakoće korištenja. Rezultati su pokazali da su metode višefaktorske i dvofaktorske autentikacije te metode pristupom treće strane najbolje za implementaciju u web aplikacije jer pružaju visoku sigurnost uz lakoću korištenja.

Ključne riječi: autentikacija, web aplikacija, sigurnost, učinkovitost, analiza

ABSTRACT

Research, development and analysis of different authentication methods in web applications

This final paper investigates and implements different authentication methods in web applications in order to examine their effectiveness and reliability in terms of user security and ease of use. The goal of the work was to find the most efficient and secure method of authentication for web applications. The research included the analysis of various authentication methods and the implementation of the most commonly used methods in a simple interface so that they could be further tested and analyzed. By implementing the selected methods and their additional analysis, it was determined which authentication methods are the best in the context of security and ease of use. The results showed that multi-factor and two-factor authentication methods and third-party access methods are the best for implementation in web applications because they provide high security with ease of use.

Keywords: authentication, web application, security, efficiency, analysis

PRILOZI

Na Gitlab-u se nalazi praktični dio: https://gitlab.com/dzejkoberi/picstream.git