Analiza ponašanja korisnika pri korištenju web aplikacija s ciljem detekcije kompromitacije korisničkih računa

Lakić, Leon

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 1471

ANALIZA PONAŠANJA KORISNIKA PRI KORIŠTENJU WEB APLIKACIJA S CILJEM DETEKCIJE KOMPROMITACIJE KORISNIČKIH RAČUNA

Leon Lakić

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 1471

ANALIZA PONAŠANJA KORISNIKA PRI KORIŠTENJU WEB APLIKACIJA S CILJEM DETEKCIJE KOMPROMITACIJE KORISNIČKIH RAČUNA

Leon Lakić

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 4. ožujka 2024.

ZAVRŠNI ZADATAK br. 1471

Pristupnik:	**Leon Lakić (0036534636)**
Studij:	Elektrotehnika i informacijska tehnologija i Računarstvo
Modul:	Računarstvo
Mentor:	izv. prof. dr. sc. Marin Vuković

Zadatak: **Analiza ponašanja korisnika pri korištenju web aplikacija s ciljem detekcije kompromitacije korisničkih računa**

Opis zadatka:

Metode autentifikacije u web aplikacijama danas variraju od tradicionalnog korištenja imena i lozinke, preko višefaktorske autentifikacije, pa sve do naprednijih metoda temeljenih na infrastrukturi javnog ključa. Međutim, malo pažnje se usmjerava na analizu ponašanja korisnika nakon uspješne autentifikacije. U tom će smislu napadač imati punu slobodu korištenja usluga ako uspije probiti prvu liniju autentifikacije.

Rok za predaju rada: 14. lipnja 2024.

Sadržaj

1. Uvod

Analizom ponašanja korisnika može se dobiti uvid na koji način korisnik koristi aplikaciju, što omogućava poboljšanje korisničkog iskustva, ali i identifikaciju potencijalno sumnjivih korisnika. Kroz prikupljanje raznih informacija o korištenju aplikacije, npr. kretanje miša[1], vrijeme provedeno na pojedinim web-lokacijama, korištenje tipkovnice te drugim parametrima, moguće je saznati uzorke ponašanja za svakog korisnika. Svaki korisnik ima jedinstven način kako pretražuje i koristi web-aplikaciju[2]. Ti uzorci korištenja pružaju temelje za autentifikaciju korisnika.

Ako napadač sazna informaciju o prvoj liniji obrane originalnog vlasnika računa, najčešće korisničko ime i lozinka, imat će punu slobodu korištenja resursa i usluga, potencijalno ih i zloupotrijebiti. Upravo zato je važna analiza ponašanja korisnika jer omogućava otkrivanje neovlaštenog pristupa koji može biti indikator napada na račun.

Ovakav način autentifikacije se razlikuje od biometrijskih metoda poput otiska prstiju i prepoznavanje lica po tome što se ne oslanja na fizički aspekt korisnika, već na način korištenja aplikacije. Na ovaj način se postiže kontinuirana autentifikacija korisnika jer se ponašanje prati tijekom cijelog vremena korištenja aplikacije.

Koristeći algoritme koji precizno analiziraju prikupljene podatke, moguće je detektirati anomalije koje odstupaju od ponašanja originalnog korisnika te poduzeti određene mjere sigurnosti. U današnje vrijeme, analiza ponašanja korisnika postaje ključni alat u razvoju web-aplikacija.

2. Povijest bihevioralne autentifikacije

Korijeni autentifikacije korisnika na temelju korištenja ulaznog uređaja seže unatrag do razdoblja Drugog svjetskog rata kada su operatori telegrafa razvili jedinstvene načine slanja Morseovog koda, kod koji predstavlja sustav točaka (.) i crtica (-) za slanje poruka. Karakteristična vremenska trajanja pritiska na tipku za slanje signala omogućili su identifikaciju operatora, čime su vojske mogle razlikovati prijateljske operatore od neprijateljskih lažnih signala. Ova praksa prepoznavanje jednostavnih obrazaca kod korištenja ulaznih uređaja postavila je temelje za modernu biometriju ponašanja.

Jedan od primjera poboljšanja sustava za bihevioralno raspoznavanje korisnika, odonosno čovjeka je implementacija sustava nakon napada na World Trade Center u New Yorku 9. rujna 2001. godine[3]. Taj napad je bio jasan indikator neuspjeha tadašnjih sigurnosnih sustava bazirani na ponašanju čovjeka. Uprava za sigurnost u prometu (TSA, Transportation Security Administration) je 2007. godine stvorila nove pozicije "Službenika za otkrivanje ponašanja" (Behavioral Detection Officers). Ovi službenici su imali zadatak procjenjivati putnike koji čekaju na sigurnosne provjere prema skupu indikatora, kao što su razine stresa i neuobičajeno ponašanje.

Pored toga, implementacija bihevioralnih biometrijskih sustava našla je primjenu i u financijskim institucijama, gdje se koriste za prepoznavanje neovlaštenih transakcija i zaštitu korisničkih računa. Analiza obrazaca korištenja, kao što su način tipkanja, brzina kretanja miša i učestalost klikova, omogućava sustavima da prepoznaju odstupanja od uobičajenog ponašanja korisnika i automatski reagiraju na potencijalne prijetnje.

Ovaj razvoj pokazuje kako su sustavi za praćenje i analizu ponašanja postali sve sofisticiraniji kroz povijest kako bi se prilagodili novim izazovima i potrebama.

3. Korištene tehnologije i alati

Za testiranje algoritama analize ponašanja korisnika napravljena je jednostavna bankovna web-aplikacija. Aplikacija omogućuje korisnicima da obavljaju bankovne operacije poput transakcija među ostalim registriranim korisnicima, pregled stanja računa te pregled povijesti obavljenih transakcija.

3.1. Korisničko sučelje

Korisničko sučelje (frontend) aplikacije je rađena u skriptnom jeziku JavaScript, a korištena je biblioteka React. React (React.js)[4] je besplatna JavaScript biblioteka otvorenog izvora koja se bazira na komponentama. Osim Reacta, korišteni su HTML i CSS za strukturiranje web stranica.

HTML (HyperText Markup Language) je osnovni jezik za izradu web stranica. Ono omgućava izradu raznih komponenti, gdje su u ovom radu korišteni obrasci, botuni, mjesta za unos podataka itd.

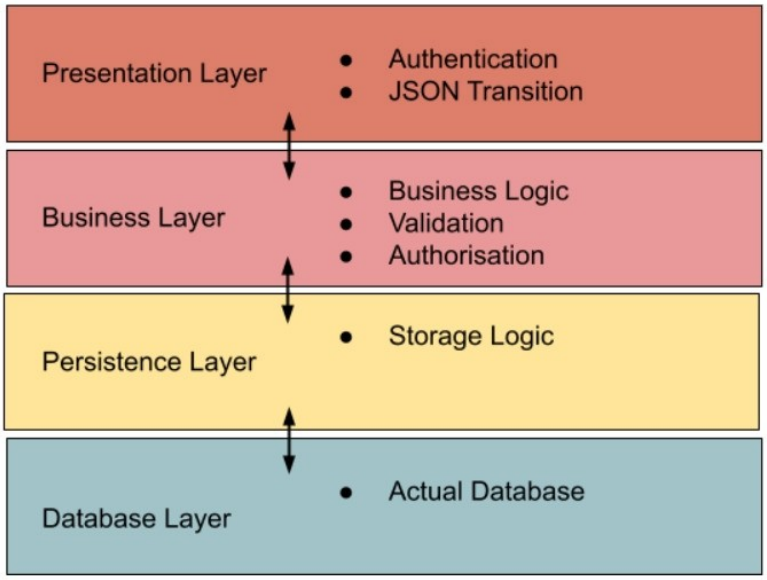
CSS (Cascading Style Sheets) je jezik za opisivanje izgleda i formatiranja HTML dokumenata. On opisuje kako će koja komponenta izgledati na web stranici uzimajući u obzir estetiku i raspored elemenata.

3.2. Pozadinsko sučelje

Pozadinsko sučelje (backend) aplikacije je rađena u programskom jeziku Java, a korišten je sustav Spring Boot. Spring Boot je alat koji omogućuje brži razvoj aplikacija pružajući gotove konfiguracije te se bazira na tri sloja: prezentacijski sloj (Controller[5]), sloj gdje se obavlja poslovna logika (Service[6]) i sloj za pristup podacima u bazi podataka (Repository[7]).

Prezentacijski sloj upravlja s korisničkim sučeljem i koristi kontrolere za obradu HTTP

zahtjeva i vraćanje odgovora.
U sloju poslovne logike se provodi logika i manipulacija nad podacima prije daljnjeg slanja istih u prezentacijski sloj ili sloju podatkovnog pristupa.
Sloj podatkovnog pristupa upravlja pristupom bazi podataka koristeći razne alate (npr. JPA (Java Persistence API) ili ORM (Object-Relational Mapping)).



Slika 3.1. Slojevi[8], Spring Boot

Korištene su navedene tehnologije koje pruža Spring Boot u sklopu razvoja bankovne web-aplikacije:

- Spring Data JPA - spremanje i dohvaćanje entiteta u relacijskim bazama podataka
- Spring Security - omogućuje dodavanje sigurnosnih značajki, uključujući autentifikaciju i autorizaciju

3.3. Baza podataka

Kao sustav za upravljanjem bazama podataka korišten je PostgreSQL. PostgreSQL je besplatan sustav za upravljanje bazama podataka koji podržava ACID (Atomicity, Consistency, Isolation, Durability) transakcije, što je ključno za osiguranje i integriteta podataka u bankovnim aplikacijama. Povezanost baze podataka s pozadinskim sučeljem omogućuje automatsko mapiranje entiteta te korištenje raznih metoda za manipulaciju spremljenim podacima bez potrebe za pisanjem SQL upita.

4. Sadržaj baze podataka

Baza podataka potrebna za ovaj sustav je strukturirana da podržava sve ključne funkcionalnosti aplikacije, uključujući korištenje same aplikacije te praćenje ponašanja korisnika.

Baza podataka ima nekoliko tablica (konkretnije četiri):

Atributi	Tip podataka	Opis
user_id	INTEGER	Primarni ključ (PK)
user_name	VARCHAR(255)	Ime korisnika
user_surname	VARCHAR(255)	Prezime korisnika
email	VARCHAR(255)	E-mail korisnika
username	VARCHAR(255)	Korisničko ime
password	VARCHAR(255)	Lozinka
created_at	VARCHAR(255)	Datum i vrijeme registracije
role	VARCHAR(255)	Uloga korisnika
balance	DECIMAL(10,2)	Stanje računa

Tablica 4.1. Struktura tablice "users"

Tablica "users" sadrži sve registrirane korisnike. Atribut "user_id" je primarni ključ ove tablice te je on jedinstven. Atribut "created_at" je u formatu "yyyy-MM-dd-HH-mm-ss" te je to vrijeme i datum zabilježeno kada se korisnik uspješno registrirao. Atribut "role" je za sve korisnike "USER". Ostali atributi su jednaki kao oni koje korisnik unese prilikom ispunjavanja obrasca za registraciju.

Atributi	Tip podataka	Opis
transaction_id	INTEGER	Primarni ključ (PK)
from_user_id	VARCHAR(255)	Korisnik pošiljatelj (FK)
to_user_id	VARCHAR(255)	Korisnik primatelj (FK)
amount	DECIMAL(10,2)	Poslani iznos
transaction_time	VARCHAR(255)	Vrijeme i datum provedene transakcije
message	VARCHAR(255)	Opcionalna poruka koja ide uz poslani iznos

Tablica 4.2. Struktura tablice "transactions"

Tablica "transactions" sadržava sve provedene transakcije ikada u aplikaciji. Atribut "transaction_id" je primarni ključ ove tablice. Atributi "from_user_id" i "to_user_id" su strani ključevi koji se referenciraju na tablice "users". Atribut "from_user_id" govori koji korisnik šalje sredstva, a "to_user_id" koji korisnik prima sredstva.

Atributi	Tip podataka	Opis
mouse_id	INTEGER	Primarni ključ (PK)
user_id	VARCHAR(255)	Korisnik (FK)
data	JSONB	Lista točaka
url	VARCHAR(255)	Mjesto gdje se odvela akcija
recorded_at	VARCHAR(255)	Datmu i vrijeme kada je zapisana akcija
mouse_type	VARCHAR(255)	Akcija miša

Tablica 4.3. Struktura tablice "mouse_movement"

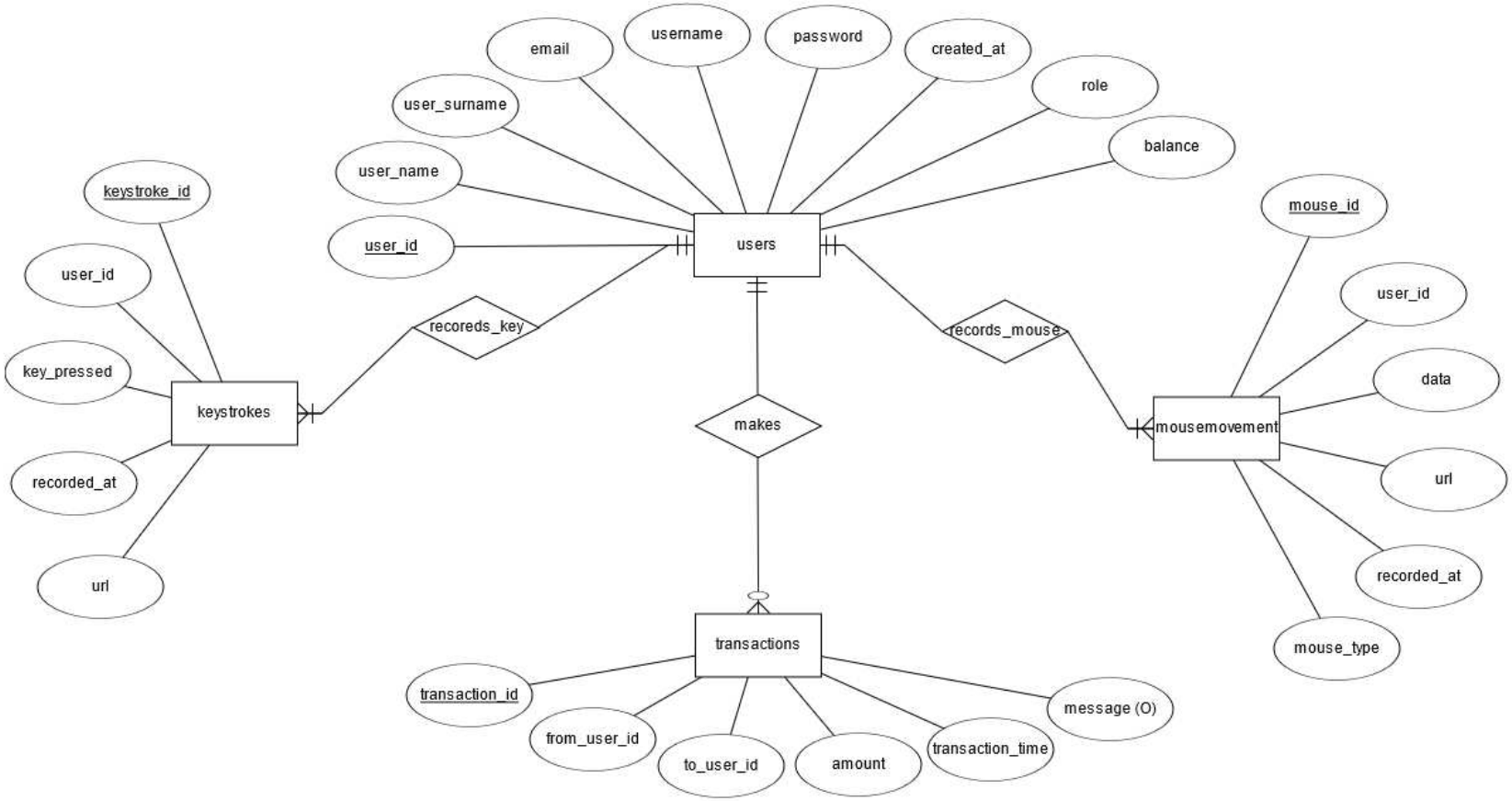
Kako je potrebno skupljati podatke o tome kako se korisnik ponaša tijekom korištenja aplikacije, uvedene su tablice "mouse_movement" i "keystrokes". "Mouse_movement" je tablica koja sadrži podatke gdje, kada i kako se miš kretao. Atribut "mouse_id" je primarni ključ ove tablice. Atribut "user_id" je strani ključ koji se referencira na tablicu "users". Atribut "data" tipa jsonb sadrži listu točaka. Te točke, koje se sastoje od x i y koordinata govore kako se miš kretao u jedinici vremena. Atribut "recorded_at" je u jednakom formatu kao i atribut "created_at" tablice "users" te on sadrži vrijeme kada se dogodio taj pomak. Atribut "mouse_type" može biti vrijednosti "mousemove", što ukazuje da se miš kretao u tom trenutku ili vrijednosti "click" što ukazuje da se pritisnuo lijevi klik miša.

Atributi	Tip podataka	Opis
keystroke_id	INTEGER	Primarni ključ (PK)
user_id	VARCHAR(255)	Korisnik (FK)
key_pressed	VARCHAR(255)	Pritisnuta tipka na tipkovnici
url	VARCHAR(255)	Mjesto gdje se pritisnula tipka
recorded_at	VARCHAR(255)	Datmu i vrijeme kada je zapisana akcija

Tablica 4.4. Struktura tablice "keystrokes"

Tablica "keystrokes" sadrži sve informacije o tome kada, kako i gdje se pritisnula neka tipka na tipkovnici. Atribut "keystroke_id" je primarni ključ. Ekvivalentno kao i kod tablice "mouse_movement", atribut "user_id" je strani ključ koji se referencira na tablicu "users". Atribut "key_pressed" daje informaciju koja tipka je pritisnuta u datom vremenu. Za razliku od atributa "recorded_at" u tablici "mouse_movement", ovdje je taj atribut

u formatu "yyyy-MM-dd-HH-mm-ss.SSS" gdje je dodana jedinica vremena milisekunda zaokružena na tri decimale zbog postizanja bolje preciznosti (razlika između pritiska jedne i druge tipke na tipkovnici često može biti manje od jedne sekunde).



Slika 4.1. ER model baze podataka

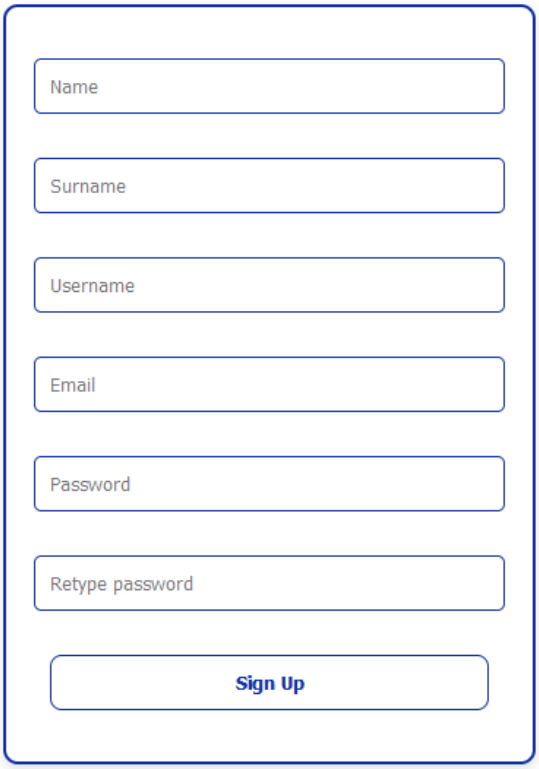
5. Funkcionalnosti aplikacije

U sklopu analize korištenja aplikacije, napravljena je jedna jednostavna web-aplikacija (nazvana "BankingApp") koja simulira funkcionalnosti neke od postojećih bankovnih aplikacija (npr. PayPal[9]). Aplikacija omogućuje registraciju i prijavu korisnika, vršenje transakcija, pregled povijesti transakcija i uvid u stanje.

5.1. "BankingApp" aplikacija

5.1.1. Registracija korisnika

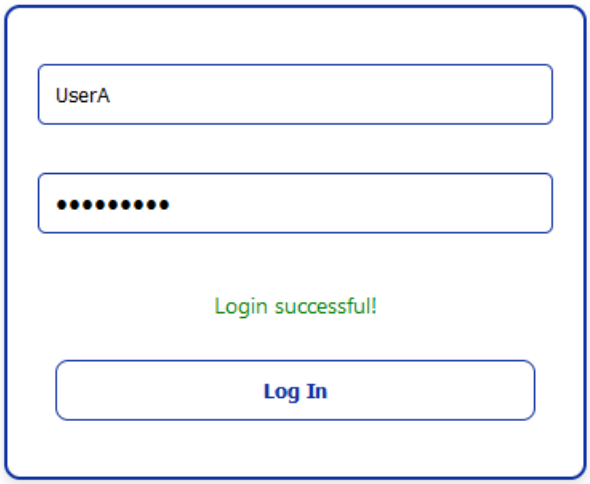
Ako se korisnik želi registrirati u sustav, mora unijeti svoje ime, prezime, korisničko ime, e-mail, lozinku te ponovljenu tu istu lozinku. Lozinka mora u sebi sadržavati barem jedno veliko slovo (ASCII vrijednosti između 65 i 90) i jedan specijalni znak.
Ako se korisnik uspješno registrirao, njegovi atributi će biti zapisani u bazu podataka u tablici "users". Biti će zapisana sva polja s obrasca i vrijeme kad se korisnik uspješno registrirao u formatu "yyyy-MM-dd-HH-mm-ss". Lozinka se enkriptira s pomoću Spring Security alata i njene klase PasswordEncoder. Za kriptiranje i zaštitu zaporke koristi se BCryptPasswordEncoder() klasa. Ona funkcionira tako da uzme samu zaporku (ono što je korisnik unio u obrazac) i generira salt, nasumični niz znakova, koji se dodaje na zaporku. Zatim takav niz (zaporka + salt) prolazi kroz jednosmjerni algoritam hashiranja. Na taj način lozinka ostaje sačuvana čak i u slučaju ako napadač dođe do same baze podataka.



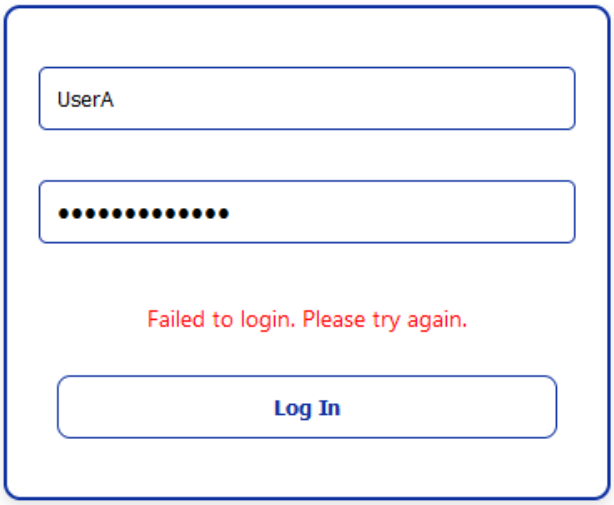
Slika 5.1. Obrazac za registriranje korisnika

5.1.2. Prijavljivanje korisnika

Nakon uspješnog registriranja u sustav, aplikacija se vraća na početnu stranicu gdje korisnik može birati između prijavljivanja u sustav i (ponovnog) registriranja. Korisnik se želi prijaviti sa svojim korisničkim imenom i lozinkom. Ako je uneseno netočno korisničko ime ili netočna lozinka, sustav će javiti korisniku da je netočan unos podataka s porukom "Failed to login. Please try again.". S druge strane, ako je korisnik unio točne podatke, sustav će mu odgovoriti s porukom "Login successful!" i nakon nekoliko sekundi će ga preusmjeriti na početnu stranicu.

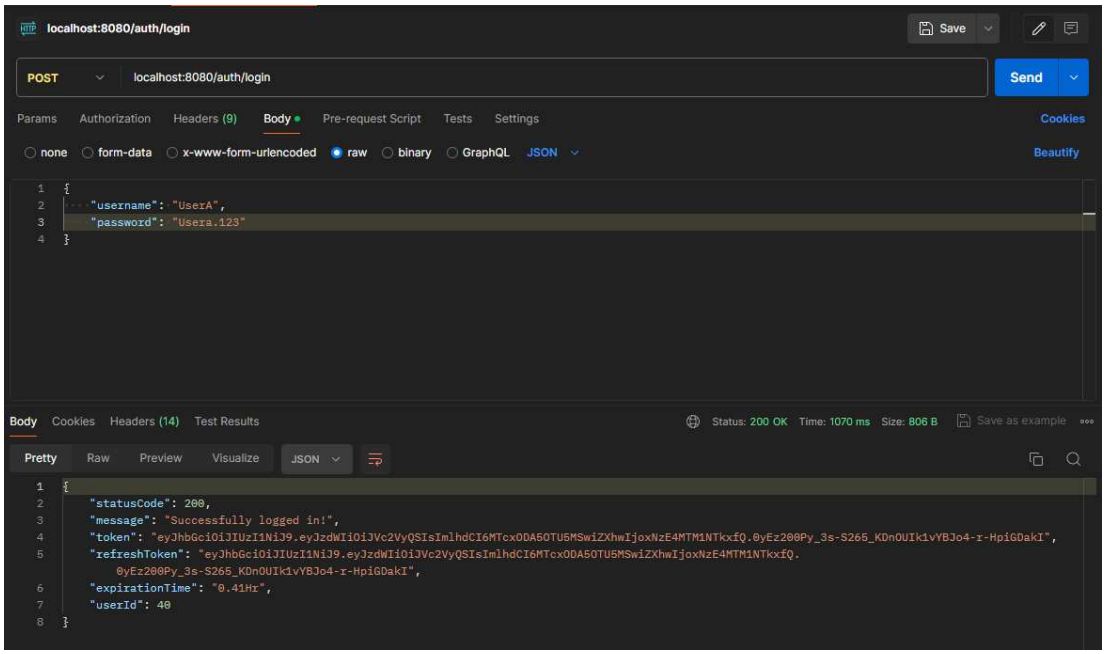


Slika 5.2. Uspješna prijava



Slika 5.3. Neuspješna prijava

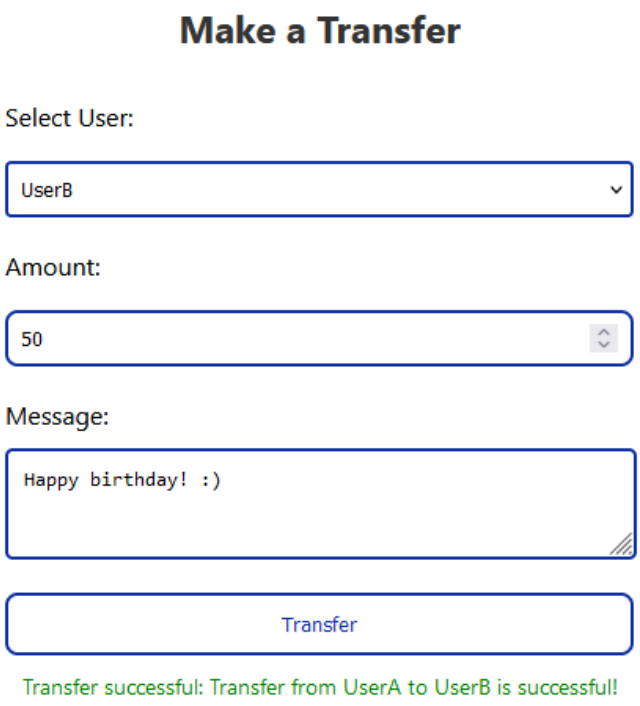
Nakon što se korisnik uspješno prijavio u sustav, dodijeljen mu je JWT (JSON Web Token) koji omogućuje autorizaciju za daljnje korištenje aplikacije. Dodijeljeni JWT se koristi za sigurno slanje podataka između klijenta i poslužitelja te osigurava da samo ovlašteni korisnici imaju pristup određenim resursima. Generirani JWT, koji ima određeno trajanje, od trenutka prijavljivanja u sustav ide uvijek uz sve HTTP zahtjeve (GET, POST, DELETE, PUT). Sustav će uvijek provjeravati valjanost JWT-a prije nego što odobri pristup resursima. S druge strane, ako se korisnik odluči odjaviti iz sustava, generirani JSON Web Token se briše.



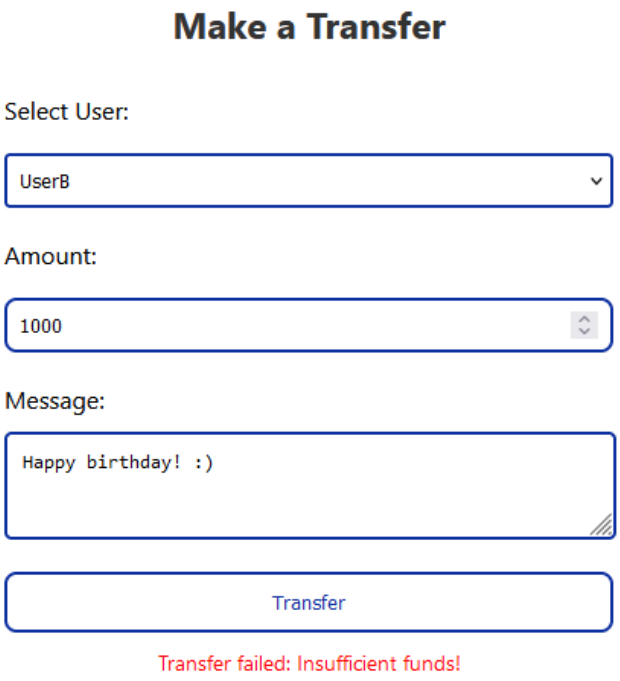
Slika 5.4. Generiranje JSON Web tokena prilikom prijavljivanja, Postman

5.1.3. Obavljanje transakcija

Korisnik ima mogućnost napraviti transakciju, odnosno poslati određenu svotu kredita nekom od već registriranih korisnika. Klikom miša na "Meni" te onda na "Do a Transaction", sustav preusmjerava korisnika na stranicu izvršavanje transakcija gdje može odabrati korisnika kojem želi poslati sredstva, unijeti iznos te po želji napisati poruku. Sustav provjerava sva polja te ovisno o uvjetima se transakcija provodi. Ako korisnik upiše iznos veći od onoga s kojim raspolaže, transakcija se neće provesti i sustav će javiti korisniku s porukom "Transfer failed: Insufficient funds!". U slučaju da su svi uvjeti zadovoljeni, transakcija će se provesti te će sustav javiti odgovarajućom porukom te sustav preusmjerava korisnika nakon nekoliko sekundi na početnu stranicu.



Slika 5.5. Uspješno provedena transakcija

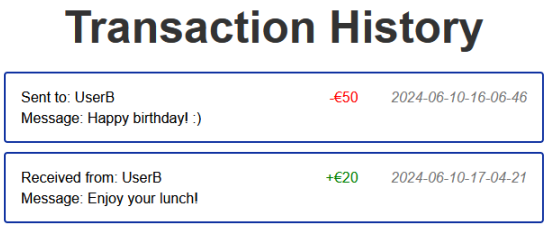


Slika 5.6. Neuspješno provedena transakcija

5.1.4. Povijest transakcija

Korisnik ima mogućnost pregleda svih transakcija koje su se dogodile otkako je napravio račun na "BankingApp" aplikaciji. Klikom miša na "Meni" te na "Previous Transactions", korisnika se preusmjerava na stranicu gdje može vidjeti povijest svih transakcija koje su izvršene preko korisnikovog računa.
Te transakcije mogu biti:

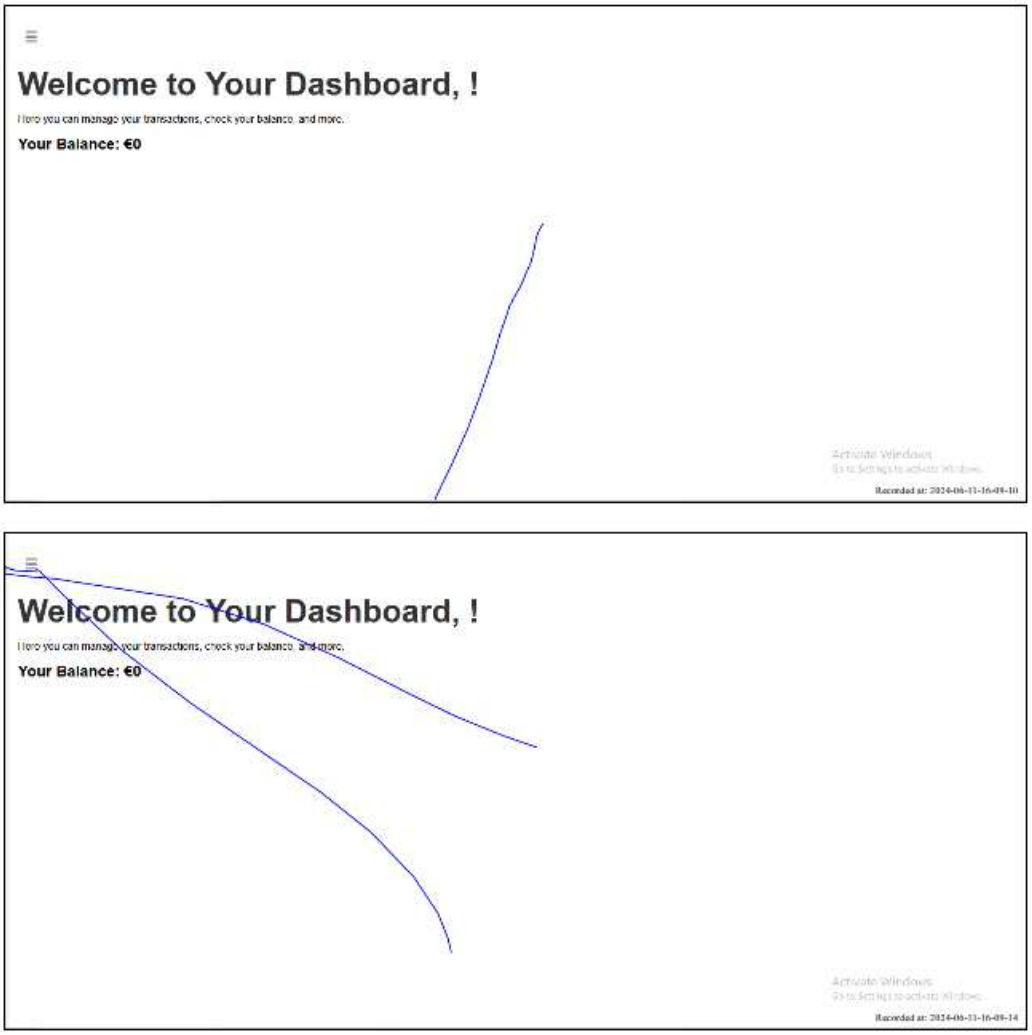
- Slanje sredstava drugim korisnicima - iznos je obojen u crvenoj boji kao znak da su se sredstva potrošila s računa
- Primanje sredstava od drugih korisnika - iznos je obojen u zelenoj boji kao znak da su se sredstva dobavljena na račun



Slika 5.7. Povijest provedenih transakcija

5.2. Program za crtanje putanje miša

Uz web-aplikaciju "BankingApp", napravljena je jednostavna "sketch.html" datoteka koja dobavlja podatke iz tablice "mouse_movement" te crta kako se miš kretao za bilo kojeg korisnika uz opciju da se postavi vrijednost donje i gornje vremenske granice. U donjem desnom kutu slike se nalazi vrijednost "recorded_at" atributa. Za svaki zapisani redak u tablici "mouse_movement" postoji jedna slika. Prvo se pokreće pozadinsko sučelje kako bi se dohvatili podatci, zatim s pomoću Live Server (alata u Visual Studio Code IDLE-u) se pokreće "sketch.html" datoteka. Preusmjerenje je na preglednik te je moguće vidjeti svaku sliku kako, gdje sve i kada se miš pomicao. Program za crtanje u slučaju pomicanje miša crta plave linije po slici, a u slučaju pritiska lijevog klika na mišu, crta se crvena točka te ona označuje gdje se taj klik dogodio na aplikaciji.



Slika 5.8. Primjer crteža za pomak miša u sekundi

Slika 5.9. Primjer crteža za klik miša

6. Prepoznavanje neovlaštenog pristupa na temelju korištenja miša

Nakon što je uvedena jednostavna bankovna web-aplikacija, sada je moguće snimanje i zapisivanje podatka ponašanja korisnika. To uključuje kako korisnik koristi miš (pomak i klik) te kako tipka na tipkovnici. Ovi podaci, koji se kontinuirano zapisuju u bazi podataka, koriste se za analizu s pomoću algoritama definiranih u pozadinskom sučelju. Cilj ovih algoritama je prepoznati neovlašteni pristup u slučaju provale. U slučaju da pozadinsko sučelje pretpostavi da je riječ o provali nečijeg računa, oslanjajući se na definirane algoritme, korisničko sučelje će javiti porukom da se radi o provali i odjaviti tog korisnika iz sustava do daljnjeg.
Svi algoritmi prvo uzimaju određeni period vremena kako bi naučili kako se korisnik ponaša. Pretpostavka je da je originalni korisnik koristio već neko vrijeme "BankingApp" aplikaciju. Nakon što je prošao određeni period učenja, algoritmi uzimaju podatke za testiranje na način koji osigurava postupni prijelaz iz faze učenja u fazu testiranja. Za ovaj postupak se koristi tehnika kliznog prozora. On omogućava kontinuiranu procjenu ponašanja korisnika čime se smanjuje rizik od naglih prijelaza i netočnih detekcija.
Kao što je prije napomenuto, akcije miša dijele se na dva tipa:

- mouse_type = 'mousemove' - u jednoj sekundi su snimljeni skup točaka gdje se miš kretao po ekranu.
  Primjer:

```
{
    "mouse_id": 567,
    "user_id": 40,
    "data": [
```

```
            {"x_coordinate": 888, "y_coordinate": 566},
            {"x_coordinate": 892, "y_coordinate": 563},
            {"x_coordinate": 898, "y_coordinate": 561},
            {"x_coordinate": 901, "y_coordinate": 559},
            {"x_coordinate": 902, "y_coordinate": 558},
            {"x_coordinate": 902, "y_coordinate": 554},
            {"x_coordinate": 901, "y_coordinate": 552},
            {"x_coordinate": 897, "y_coordinate": 543},
            {"x_coordinate": 865, "y_coordinate": 507},
            {"x_coordinate": 807, "y_coordinate": 453},
            {"x_coordinate": 743, "y_coordinate": 400},
            {"x_coordinate": 664, "y_coordinate": 347},
            {"x_coordinate": 583, "y_coordinate": 290},
            {"x_coordinate": 488, "y_coordinate": 226},
            {"x_coordinate": 345, "y_coordinate": 131},
            {"x_coordinate": 253, "y_coordinate": 69},
            {"x_coordinate": 178, "y_coordinate": 22},
            {"x_coordinate": 151, "y_coordinate": 5}
        ],
        "url": "http://localhost:3000/user/40",
        "recorded_at": "2024-06-11-16-10-44",
        "mouse_type": "mousemove"
    }
```


- mouse_type = ’click’ - u bilom kojem trenutku zapisivanja informacija, ako se dogodi da je korisnik pritisnuo lijevi klik na mišu, sustav će isti taj trenutak zapisati u bazu podataka


```
    {
        "mouse_id": 569,
        "user_id": 40,
        "data": ["x_coordinate": 27, "y_coordinate": 46],
```

```
        "url": "http://localhost:3000/user/40",
        "recorded_at": "2024-06-11-16-10-49",
        "mouse_type": "click"
    }
```

6.1. Pokreti miša

Pokreti miša predstavljaju ključan aspekt biometrije ponašanja u analizi korisničkog ponašanja[10]. Uzimajući u obzir brzinu miša (prijeđeni put u jedinici vremena), smjer kretanja, intenzitet promjene smjera i akcije koje su se koristile s mišom, moguće je identificirati karakteristike svakog korisnika.
Svi parametri koji algoritam uzima u obzir za prepoznavanje neovlaštenog pristupa koriste aritmetičku sredinu i standardnu devijaciju. Ako se dobivena vrijednost parametra nalazi u određenom intervalu, aplikacija nastavlja normalno s radom i ne detektira nikakvu anomaliju. S druge strane, ako se u kontinuiranoj provjeri neki od parametara nađe izvan određenog intervala, sustav prepoznaje anomaliju. Interval je definiran kao tri puta standardna devijacija oduzeta od, odnosno zbrojena s aritmetičkom sredinom:

$$(\bar{x} - 3\sigma, \bar{x} + 3\sigma)$$

Aritmetička sredina:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Standardna devijacija:

$$\sigma = \sqrt{\frac{1}{N-1}\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

6.1.1. Brzina kretanja

Prvi parametar koji algoritam uzima u obzir jest brzina kretanja miša. Kako miš putuje po ekranu svaki put kada ga korisnik pomake, on se giba po točkama koje čine putanju. Svaka točka sadrži x i y koordinatu. Sustav u jednoj sekundi zapiše u tablicu "mo-

use_movement" listu točaka, odnosno listu gdje su elementi parovi x i y koordinata. Za izračun puta, koristi se formula Euklidske udaljenosti[11] između dviju točaka:

$$d(A,B) = \sqrt{(x_B - x_A)^2 + (y_B - y_A)^2}$$

Nije nužno da će u jednoj sekundi biti zapisane samo dvije točke, stoga treba uzeti u obzir sve točke. Euklidska udaljenost između dvaju vektora $\mathbf{p} = (p_1, p_2, \dots, p_n)$ i $\mathbf{q} = (q_1, q_2, \dots, q_n)$ može se izračunati koristeći sljedeću formulu:

$$s = d(\mathbf{p}, \mathbf{q}) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

```
1 usage
private double calculateDistance(MouseMovement mouseMovement) {
    double xDiff;
    double yDiff;
    double totalDistance = 0.f;
    if(mouseMovement.getMouse_type().equals("mousemove")) { //ignore the clicks
        for(int i = 0; i < mouseMovement.getData().size(); i++) {
            if(i != mouseMovement.getData().size() - 1) {
                xDiff = mouseMovement.getData().get(i + 1).getX_coordinate()
                        - mouseMovement.getData().get(i).getX_coordinate();
                yDiff = mouseMovement.getData().get(i + 1).getY_coordinate()
                        - mouseMovement.getData().get(i).getY_coordinate();

                //distance = sqrt(x^2 + y^2)
                totalDistance = totalDistance + Math.sqrt(Math.pow(xDiff, 2) + Math.pow(yDiff, 2));
            }
        }
    }
    return totalDistance;
}
```

Slika 6.1. Metoda za izračun prijeđenog puta

Kako sustav definirano zapisuje putanju svaki sekund, brzina je jednaka iznosu te zapisane putanje u tom datom vremenu i trenutku.

$$v = \frac{s}{t}$$

Kako vrijedi $t = 1$ s:

$$v = \frac{s}{1\,\text{s}} = s\,[\text{mjernih jedinica/s}]$$

6.1.2. Stupanj zakrivljenosti putanje

Pojedini korisnici se razlikuju po tome koliko brzo, odnosno naglo mijenjaju smjer putanje miša. Neki korisnici češće rade glatke pokrete mišom, dok neki rade grčevite pokrete. Mjerenje stupnja zakrivljenosti putanje može pomoći u prepoznavanju takvih razlika. Algoritam će uzeti tri uzastopne točke (npr. A, B i C) i izračunati kut između dužina $\overline{AB}$ i $\overline{BC}$.

Definiramo vektore **a** i **b**:

$$\mathbf{a} = (x_B - x_A, y_B - y_A)$$

$$\mathbf{b} = (x_C - x_B, y_C - y_B)$$

Formula skalarnog umnoška:

$$\mathbf{a} * \mathbf{b} = \mathbf{b} * \mathbf{a} = |\mathbf{a}| * |\mathbf{b}| * \cos(\theta)$$

Kut ostaje sam na jednoj strani jednadžbe i formula za izračun kuta glasi:

$$\theta = \arccos(\frac{\mathbf{a} * \mathbf{b}}{|\mathbf{a}| * |\mathbf{b}|})$$

Ova mjerenja zakrivljenosti mogu pomoći u razlikovanju korisnika na temelju intenziteta promjene strane miša.

```
1 usage
private double calculateAngleBetweenVectors(Coordinates dotA, Coordinates dotB, Coordinates dotC) {
    double x_1 = dotB.getX_coordinate() - dotA.getX_coordinate();
    double x_2 = dotC.getX_coordinate() - dotB.getX_coordinate();
    double y_1 = dotB.getY_coordinate() - dotA.getY_coordinate();
    double y_2 = dotC.getY_coordinate() - dotB.getY_coordinate();

    double product = (x_1 * x_2) + (y_1 * y_2);
    double length1 = Math.sqrt(Math.pow(x_1, 2) + Math.pow(y_1, 2));
    double length2 = Math.sqrt(Math.pow(x_2, 2) + Math.pow(y_2, 2));

    return Math.acos(product / (length1 * length2));
}
```

Slika 6.2. Metoda za izračun kuta

7. Prepoznavanje neovlaštenog pristupa na temelju korištenja tipkovnice

Korištenje tipkovnice predstavlja još jedan ključan aspekt biometrije ponašanja u analizi korisničkog ponašanja[12]. Svaki korisnik ime jedinstven način tipkanja koji uključuje razliku vremena između pritiska na dvije različite tipke te trajanje pritiska na tipku. Ovi parametri čine jedinstvene podatke koje je nemoguće replicirati. Uveden je parametar koji prati koliko vremena treba u jedinici vremena milisekunda prelazak s jedne tipke na tipkovnici na drugu tipku.

Kako bi se pratio određeni par tipki, uvedena je klasa KeyPair s atributima keyOne, keyTwo (oba tipa String). Također kako bi se pratilo vrijeme prelaska s dvije različite tipke, uvedena je mapa kao struktura podataka, gdje je ključ KeyPair dvojka, a vrijednost lista List<Double> sva vremena vezane za taj par tipki.

```
7 usages
public static class KeyPair {
    3 usages
    private String keyOne;
    3 usages
    private String keyTwo;
    1 usage
    public KeyPair(String keyOne, String keyTwo) {
        this.keyOne = keyOne;
        this.keyTwo = keyTwo;
    }
    no usages
    public String getKeyOne() {
        return keyOne;
    }
    no usages
    public void setKeyOne(String keyOne) {
        this.keyOne = keyOne;
    }
    no usages
    public String getKeyTwo() {
        return keyTwo;
    }
    no usages
    public void setKeyTwo(String keyTwo) {
        this.keyTwo = keyTwo;
    }
}
```

Slika 7.1. Klasa KeyPair

Na primjer, ako u tablici "keystrokes" postoji više prijelaza iz tipke "k" na tipku "o", ključ će biti ta dvojka te sva vremena izračunata kao razlika atributa "recorded_at" kao vrijednost te mape.

Nakon što je sustav "naučio" kako originalni korisnik barata s tipkama na tipkovnici, moći će to naučeno znanje primijeniti na detektiranje potencijalnih neovlaštenih pristupa.

1	173	41	k	2024-06-11-17-42-55.888	http://localhost:3000/user/41
2	174	41	o	2024-06-11-17-42-56.044	http://localhost:3000/user/41
3	175	41	l	2024-06-11-17-42-56.215	http://localhost:3000/user/41
4	176	41	i	2024-06-11-17-42-56.360	http://localhost:3000/user/41
5	177	41	k	2024-06-11-17-42-56.526	http://localhost:3000/user/41
6	178	41	o	2024-06-11-17-42-56.700	http://localhost:3000/user/41
7	179	41		2024-06-11-17-42-56.889	http://localhost:3000/user/41

Slika 7.2. Primjer zapisa u bazi podataka tijekom tipkanja riječi "koliko"

8. Usporedba dvaju korisnika

Dva korisnika su koristili web-aplikaciju i u pozadini je program zapisivao svaku njihovu radnju s mišem i tipkovnicom. Analizirajući njihove zapisane podatke ponašanja, može se napraviti usporedba koja će pokazati razlike u načinu korištenja aplikacije.
Kako bi se napravila usporedba, oba korisnika su dobili nekoliko zadataka koje su trebali izvršiti unutar aplikacije.

8.1. Usporedbe kod korištenja tipkovnice

Analizirat će se koliko brzo je svaki korisnik koristio tipkovnicu, odnosno koliko vremena je trebalo da se pređe s jedne tipke na drugu. Zadatak je bio da korisnik napravi transakciju drugom korisniku, te da napiše poruku "Happy birthday!" koja ide uz određeni iznos.

Znak	**Vrijeme stisnute tipke**	**Δt [s]**
H	16:30:10.356	0.000
a	16:30:10.523	0.167
p	16:30:10.780	0.257
p	16:30:10.918	0.138
y	16:30:11.000	0.082
spacebar	16:30:11.112	0.112
b	16:30:11.528	0.416
i	16:30:11.760	0.232
r	16:30:11.829	0.069
t	16:30:11.905	0.076
h	16:30:11.945	0.040
d	16:30:12.132	0.187
a	16:30:12.271	0.139
y	16:30:12.402	0.131
!	16:30:12.627	0.225

Tablica 8.1. Zapisano vrijeme korisniku "UserA"

U tablici je zapisano koja je tipka kada stisnuta (format HH:mm:ss.SSS, nisu uzeti u obzir dan, mjesec i godina) te razlika vremena između dva slova.
Prvom korisniku, "UserA" trebalo je 0.664 sekundi da napiše niz znakova "Happy". Trebalo mu je 1.515 sekundi da napiše niz znakova " birthday!". Dakle, cijelu poruku je napisao u roku 2.179 sekundi.

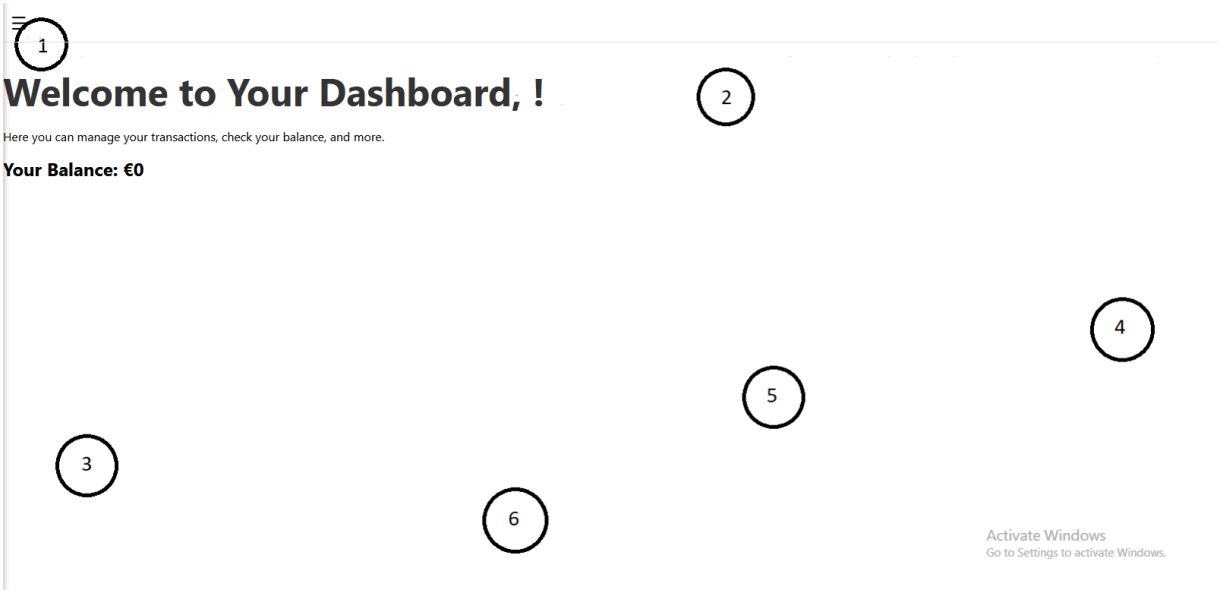
Znak	**Vrijeme stisnute tipke**	**Δt [s]**
H	16:35:52.457	0.000
a	16:35:53.140	0.683
p	16:35:53.543	0.403
p	16:35:53.843	0.300
y	16:35:54.594	0.751
spacebar	16:35:54.852	0.258
b	16:35:55.200	0.348
i	16:35:55.579	0.379
r	16:35:55.832	0.253
t	16:35:56.124	0.292
h	16:35:56.321	0.197
d	16:35:56.693	0.372
a	16:35:56.982	0.289
y	16:35:57.232	0.250
!	16:35:57.507	0.275

Tablica 8.2. Zapisano vrijeme korisniku "UserB"

Korisniku "UserB" trebalo je 2.395 sekunde da napiše riječ "Happy", što je 3.21 puta više potrebnog vremena od korisnika "UserA". Za niz znaknova " birthday!" trebalo je korisniku "UserB" 2.655 sekundi da uspješno napiše navedeni niz. To je 1.75 puta više vremena potrebno od korisnika "UserA". Na kraju, ova akcija je trajala ukupno 5.05 sekundi, 2.31 puta duže od prvog korisnika.
Najbrži prijelaz između dviju tipki kod korisnika "UserA" je t -> h, koja je trajao samo 0.040 sekundi, dok je najbrži prijelaz za korisnika "UserB" također t -> h, ali trajao je 0.197 sekundi. Najduži prijelaz je za korisnika "UserA" je spacebar -> b koji je trajao 0.416 sekundi. S druge strane, za korisnika "UserB" je najduži prijelaz p -> y koji je trajao 0.751 sekundi.
Ovi rezultati pokazuju da korisnik "UserB" ima znatno sporije vrijeme reakcije i prijelaza između dviju tipke na tipkovnici u odnosu na korisnika "UserA". Također, korisnik "UserA" je više konzistentniji jer vremena ne odmiču toliko od aritmetičke sredina u odnosnu na korisnika "UserB".

8.2. Usporedbe kod korištenja miša

Šest točaka je pozicionirano negdje na ekranu te će svaki od korisnika morati pomaknuti miš preko zadanih točaka. Korisnik mora prijeći i kliknuti lijevi klik miša u onome trenutku kada dođe do određene točke te treba poštivati redoslijed kojima će graditi put. Nakon što je zadatak obavljen, vrši se analiza podataka koja je sakupljena u tom periodu te se računa brzina kretanja od jedne do druge točke i stupanj zakrivljenosti za svakog korisnika.

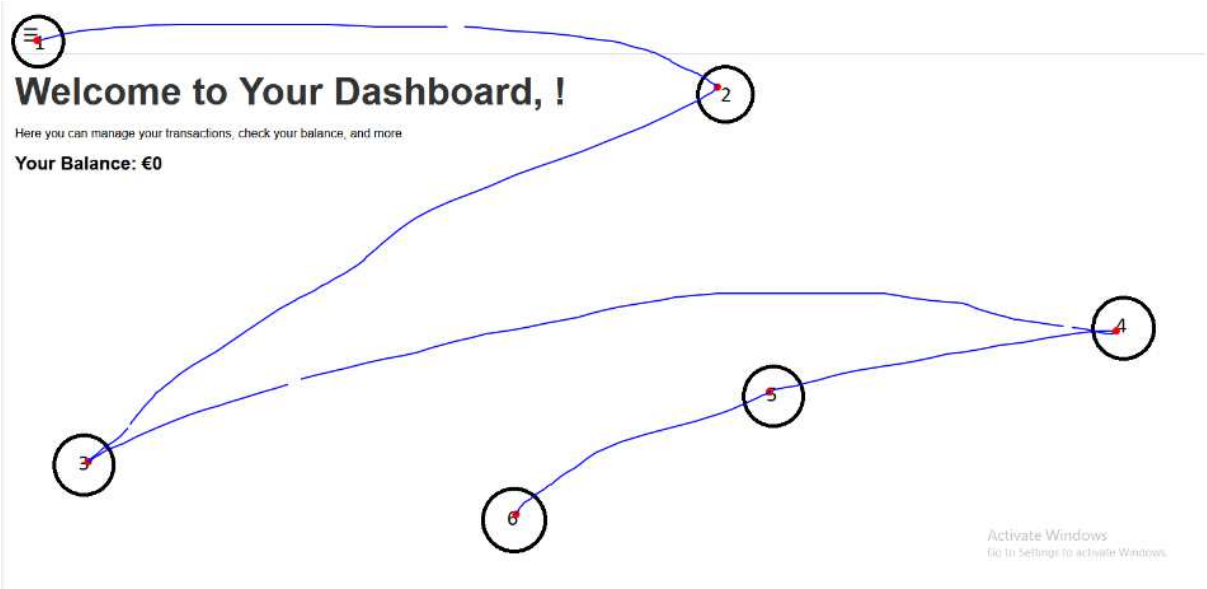


Slika 8.1. Zadane točke za put miša

Veličina liste točaka	Vrijeme	Akcija	Opis
1	17:30:05	lijevi klik	pritisak lijevog klika miša na točku 1
38	17:30:06	pomak	pomak miša od točke 1 do točke 2
43	17:30:07	pomak	pomak miša od točke 1 do točke 2
1	17:30:07	lijevi klik	pritisak lijevog klika miša na točku 2
54	17:30:08	pomak	pomak miša od točke 2 do točke 3
1	17:30:08	lijevi klik	pritisak lijevog klika miša na točku 3
31	17:30:09	pomak	pomak miša od točke 3 do točke 4
60	17:30:10	pomak	pomak miša od točke 3 do točke 4
1	17:30:10	lijevi klik	pritisak lijevog klika miša na točku 4
55	17:30:11	pomak	pomak miša od točke 4 do točke 5
1	17:30:11	lijevi klik	pritisak lijevog klika miša na točku 5
36	17:30:12	pomak	pomak miša od točke 5 do točke 6
1	17:30:12	lijevi klik	pritisak lijevog klika miša na točku 6

Tablica 8.3. Tablica o prijeđenom putu miša za korisnika "UserA"

Prvi korisnik je prešao cijeli put mišem u roku od 7 sekundi (razlika vremena između pritiska lijevog klika na točki 6 i točki 1). Promjena smjera miša na točkama je vrlo grčevita, što se posebno može primijetiti na točkama 2, 3 i 4.

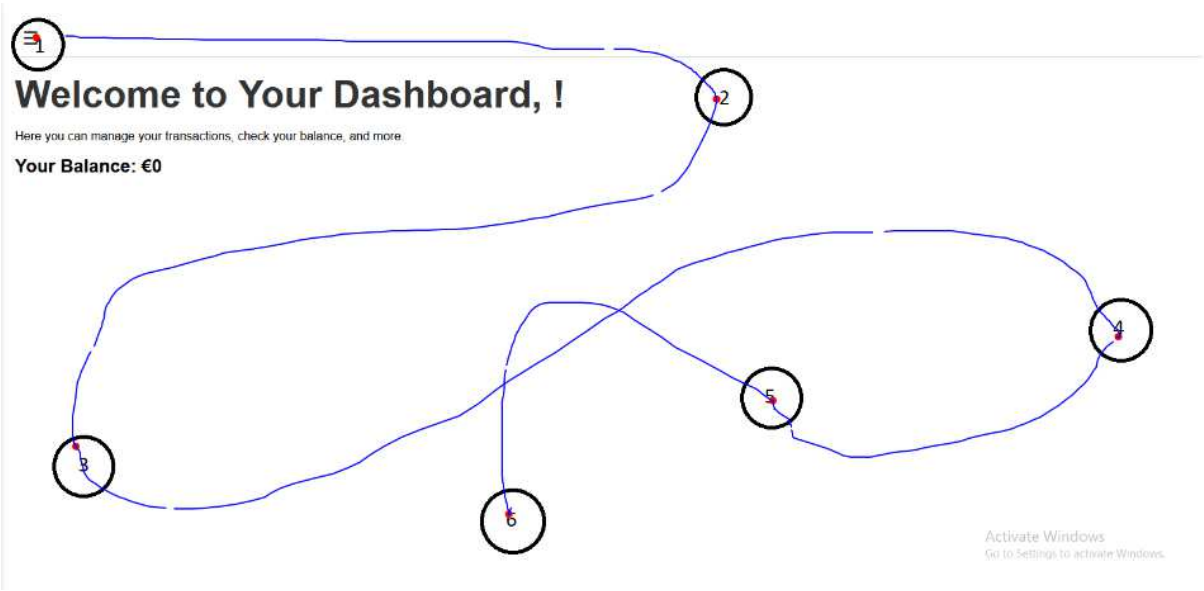


Slika 8.2. Pomak miša po točkama za korisnika "UserA"

S druge strane, drugom korisniku "UserB" trebalo je 10 sekundi da napravi put mišem preko navedenih točaka. Promjena smjera miša na točkama je blaga, bez naglih pokreta.

Veličina liste točaka	Vrijeme	Akcija	Opis
1	17:35:06	lijevi klik	pritisak lijevog klika miša na točku 1
67	17:35:07	pomak	pomak miša od točke 1 do točke 2
10	17:35:08	pomak	pomak miša od točke 1 do točke 2
1	17:35:08	lijevi klik	pritisak lijevog klika miša na točku 2
23	17:35:08	pomak	pomak miša od točke 2 do točke 3
52	17:35:09	pomak	pomak miša od točke 2 do točke 3
1	17:35:10	lijevi klik	pritisak lijevog klika miša na točku 3
43	17:35:10	pomak	pomak miša od točke 3 do točke 4
62	17:35:11	pomak	pomak miša od točke 3 do točke 4
22	17:35:12	pomak	pomak miša od točke 3 do točke 4
1	17:35:12	lijevi klik	pritisak lijevog klika miša na točku 4
60	17:35:13	pomak	pomak miša od točke 4 do točke 5
54	17:35:14	pomak	pomak miša od točke 4 do točke 5
1	17:35:14	lijevi klik	pritisak lijevog klika miša na točku 5
53	17:35:15	pomak	pomak miša od točke 5 do točke 6
1	17:35:16	lijevi klik	pritisak lijevog klika miša na točku 6

Tablica 8.4. Tablica o prijeđenom putu miša za korisnika "UserB"



Slika 8.3. Pomak miša po točkama za korisnika "UserB"

Analizom se zaključuje kako je prvi korisnik brže prešao put mišem, koristeći nagle promjene smjera na točkama. Drugom korisniku je trebalo tri sekunde duže nego prvom korisniku da prijeđe put, ali je radio blage okrete s mišom. Razlika u vremenu i stupnju zakrivljenosti na svakoj točki između ova dva korisnika jasno pokazuje razlike u njihovim ponašanjima i korištenja miša.

9. Mogući razvoj sustava

Jedan od načina kako bi se ovaj sustav mogao unaprijediti kako bi davao točnije rezultate jest implementacija modela strojnog učenja. Uvođenjem modela strojnog učenja, sustav bi mogao preciznije prepoznati obrasce u ponašanju korisnika i bolje razlikovati legitimne korisnika od napadača. Na primjer, jedan od vrste strojnog učenja koji bi bio prigodan za ovaj problem jest nenadzirano učenje. Kod nenadziranog učenja (engl. supervised learning) dani su podaci bez ciljne vrijednosti te treba naći pravilnost u podacima. Grupiranje (engl. clustering) kao jedna od varijanti nenadziranog učenja bi mogao biti izuzetno koristan kod raspoznavanja ponašanja korisnika. Koristeći algoritme za grupiranje, sustav može automatski otkriti obrasce ponašanja bez potrebe za prethodnim označavanjem podataka. Ovi algoritmi mogu grupirati slične obrasce ponašanja u klastere, što omogućava identifikaciju anomalija ili odstupanja od uobičajenog ponašanja.

Moguće je koristiti i algoritme nadziranog učenja gdje ono uzima skup podataka u obliku (ulaz, izlaz) = (x, y) te treba naći određeno preslikavanje. Primjer, koristeći neuronske mreže može se napraviti model koji predviđa vjerojatnost da je određeno ponašanje sumnjivo.

Implementacijom naprednijih algoritama strojnog učenja, sustav bi bio precizniji kod pokušaja detekcije neovlaštenog pristupa čime bi se smanjio rizik od napada.

10. Zaključak

Korištenje biometrije ponašanja u analizi korisničkog ponašanja predstavlja značajan korak naprijed u području sigurnosti i autentifikaciji korisnika. Ovaj pristup omogućava identifikaciju korisnika na temelju korištenja miša i tipkovnice, što dodaje dodatni sloj sigurnosti u zaštiti korisničkih računa. Osim analize ponašanja na web-aplikacija, postoje i mobilne verzije sustava koje prate ponašanje preko zaslona osjetljivog na dodir.
Iako ovaj sustav može biti vrlo učinkovit, valja uzeti u obzir potencijalne izazove. Prvo, ponašanje korisnika može varirati zbog mnogih faktora, kao što su raspoloženje, stres, umor ili nekakve promjene u okolini. Takvi faktori utječu na ponašanje korisnika te bi sustav mogao pogrešno zaključiti da je riječ o neovlaštenom pristupu aplikaciji. S druge strane postoji šansa da će napadač imati slično ponašanje kao i originalni korisnik te ga sustav neće otkloniti iz sustava.
Unatoč potencijalnim izazovima, biometrija ponašanja ima veliki potencijal za unapređenje sigurnosti digitalnih sustava. Razvoj naprednijih algoritama i sakupljanje veće količine podataka omogućit će povećanje preciznosti i pouzdanosti ovih metoda.

Literatura

[1] A. Cybersecurity, “What is behavioral biometrics authentication?” https://cybersecurity.asee.io/blog/what-is-behavioral-biometrics-authentication/, 2024., [Članak o biometriji ponašanja].

[2] CogniFit, “The seven learning styles: Visual learning style”, https://blog.cognifit.com/learning-styles/, [Članak o jedinstvenosti učenja].

[3] H. I. Review, “Behavioral profiling and the biometrics of intent”, https://hir.harvard.edu/behavioral-profiling-and-the-biometrics-of-intent/, [Primjeri uvođenja dodatnih sigurnosnih mjera temeljena na ponašanju korisnika].

[4] React, “Introduction to react”, https://legacy.reactjs.org/.

[5] Baledung, “Quick guide to spring controllers”, https://www.baeldung.com/spring-controllers, 2024., [Vodič o kontrolerima].

[6] GeeksForGeeks, “Spring @service annotation with example”, https://www.geeksforgeeks.org/spring-service-annotation-with-example/, 2022., [Primjer klase Service].

[7] DigitalOcean, “Spring @repository annotation”, https://www.digitalocean.com/community/tutorials/spring-repository-annotation, 2022., [Uvod u Repository].

[8] https://www.naukri.com/code360/library/spring-boot-architecture, [Spring Boot slojevi].

[9] PayPal, “What is paypal and how does it work?” https://www.paypal.com/us/digital-wallet/how-paypal-works, [Kratko objašnjenje o aplikaciji PayPal].

[10] TypingDNA, “What is mouse dynamics and how does it work in continuous authentication?” https://www.typingdna.com/glossary/what-is-mouse-dynamics-and-how-it-works, [Objašnjenje o kontinuiranoj provjeri preko višestrukog pokreta miša].

[11] Wikipedia, “Euclidian distance”, https://en.wikipedia.org/wiki/Euclidean_distance, [Formula za računanje Euklidske udaljenosti].

[12] J. Griffin, “Static keystroke dynamics”, https://www.fleksy.com/blog/keystroke-dynamics-and-the-types-of-behavioural-biometrics/#Static_Keystroke_Dynamics, 2022., [Princip pritiska tipki u sekvenci].

Sažetak

Analiza ponašanja korisnika pri korištenju web aplikacija s ciljem detekcije kompromitacije korisničkih računa

Leon Lakić

Ovaj rad se bavi analizom ponašanja korisnika u svrhu autentifikacije na web aplikacijama. Napravljena je jednostavna transakcijska web-aplikacija preko koje se zapisuju ponašanja korisnika, kako se miš pomiče i kako se tipkovnica koristi. Napravljeni su algoritmi koji uzimaju snimljene podatke i pokušavaju detektirati neovlašteni pristup u web-aplikaciju. Ako se utvrdi da je riječ o neovlaštenom pristupu, sustav generira upozorenje. Konačno, zaključeno je kako ova vrsta sustava za autentifikaciju korisnika predstavlja obećavajuću tehnologiju za budućnost, danas postoje nekoliko faktora kako bi ovaj sustav mogao donijeti i krivu procjenu.

Ključne riječi: ponašanje; miš; tipkovnica; React; Spring; PostgreSQL; napad; sigurnost; autentifikacija;

Abstract

User behavior analysis when using web applications with the aim of detecting compromise of user accounts

Leon Lakić

This paper deals with the analysis of user behavior for the purpose of authentication on web applications. A simple transactional web application was created to record user behaviors, including mouse movements and keyboard usage. Algorithms have been developed to take the recorded data and attempt to detect unauthorized access to the web application. If it is determined that there is unauthorized access, the system generates an alert. Ultimately, it is concluded that this type of user authentication system represents a promising technology for the future. However, there are several factors today that could lead this system to make incorrect assessments.

Keywords: behavior; mouse; keyboard; React; Spring; PostgreSQL; attack; security; authentication;