VIŠJA STROKOVNA ŠOLA ACADEMIA

MARIBOR

RAZVOJ NAMIZNE APLIKACIJE ZA BLAGAJNIŠKO POSLOVANJE IN POVEZAVO S FURS

Kandidat: Matej Gyergyek

Vrsta študija: študent izrednega študija

Študijski program: Računalništvo

Mentor predavatelj: mag. Dušan Brglez

Mentorica v podjetju: Urška Mesarič, mag. inž. inf. in teh. kom.

Lektor: Špela Fele, univ. dipl. slov.

Maribor, 2025

IZJAVA O AVTORSTVU DIPLOMSKEGA DELA

Podpisani/a Matej Gyergyek, sem avtor diplomskega dela z naslovom »Razvoj namizne aplikacije za blagajniško poslovanje in povezavo s FURS«, ki sem ga napisal/a pod mentorstvom »mag. Dušan Brglez«.

S svojim podpisom zagotavljam, da:

- je predloženo delo izključno rezultat mojega dela,
- sem poskrbel/a, da so dela in mnenja drugih avtorjev, ki jih uporabljam v predloženi nalogi, navedena oz. citirana skladno s pravili Višje strokovne šole Academia Maribor,
- se zavedam, da je plagiatorstvo – predstavljanje tujih del oz. misli, kot moje lastne kaznivo po Zakonu o avtorski in sorodnih pravicah (Uradni list RS, št. 16/07 – uradno prečiščeno besedilo, 68/08, 110/13, 56/15 in 63/16 – ZKUASP); prekršek pa podleže tudi ukrepom Višje strokovne šole Academia Maribor skladno z njenimi pravili,
- skladno z 32.a členom ZASP dovoljujem Višji strokovni šoli Academia Maribor objavo diplomskega dela na spletnem portalu šole.

Murski Črnci, november 2025 Podpis študenta:

ZAHVALA

Zahvaljujem se mentorjema, mag. Dušanu Brglezu in mag. inž. inf. in teh. kom. Urški Mesarič, za strokovno pomoč in vodenje pri izdelavi diplomskega dela.

Posebna zahvala gre tudi mojim staršem za podporo in spodbudo med študijem.

"Najpomembnejša motivacija za delo v šoli in življenju je veselje do dela in rezultata ter vedeti, kaj to pomeni za skupnost."

Albert Einstein

POVZETEK

Diplomsko delo obravnava razvoj sodobne namizne aplikacije za blagajniško poslovanje z integrirano povezavo s sistemom FURS za davčno potrjevanje računov. Motivacija za raziskavo izhaja iz opažanj, da številna podjetja še vedno uporabljajo zastarele ali preveč zapletene blagajniške rešitve, ki otežujejo delo uporabnikom in povečujejo možnost napak.

Teoretični del predstavlja pregled globalnega trga sistemov POS, ki je leta 2024 dosegel vrednost 17 bilijonov USD, in primerjavo lokalnih in oblačnih rešitev. Podrobno so obravnavani slovenski zakonodajni okvir za davčno potrjevanje računov preko sistema FURS, analiza varnostnih standardov (ISO/IEC 27001:2022, ISO/IEC 27002:2022, GDPR, NIS2, ZInfV-1) in načela sodobne uporabniške izkušnje v poslovnih aplikacijah.

Praktični del opisuje celovit razvoj aplikacije z uporabo tehnologij .NET 8.0, Avalonia UI Framework, Entity Framework Core in Microsoft SQL Server. Implementirani so Clean Architecture z jasno ločenimi plastmi (predstavitvena, poslovna logika, podatkovna), arhitekturni vzorec MVVM in Repository Pattern za dostop do podatkov. Posebna pozornost je namenjena varnostnim mehanizmom, vključno z BCrypt hashiranjem gesel, skladnostjo GDPR pri minimizaciji osebnih podatkov, zaščiteno komunikacijo HTTPS/TLS in preprečevanjem napadov injekcije SQL.

Aplikacija omogoča hitro in intuitivno uporabo z jasno strukturiranim uporabniškim vmesnikom, avtentikacijo uporabnikov z role-based dostopom, upravljanje izdelkov in zalog, procesiranje plačil ter simulacijo davčnega potrjevanja preko okolja FURS sandbox. Razvita rešitev predstavlja stroškovno učinkovito alternativo dragim komercialnim sistemom, še posebej za mala podjetja, samostojne podjetnike in gostinske lokale. Izvedena anketa med uporabniki sistemov POS je pokazala, da uporabniki visoko vrednotijo intuitivnost vmesnika, zanesljivost integracije FURS in nizke stroške vzdrževanja lokalnih rešitev v primerjavi z mesečnimi naročninami oblačnih sistemov.

Ključne besede: *sistem POS, blagajniška aplikacija, integracija FURS, informacijska varnost, namizna aplikacija*

ABSTRACT

Development of a Desktop Application for Point-of-Sale Operations and Integration with FURS

This thesis addresses the development of a modern desktop application for point-of-sale operations with integrated connection to the FURS system for fiscal receipt verification. The motivation for this research stems from observations that many companies still use outdated or overly complex POS solutions that hinder user workflow and increase the likelihood of errors.

The theoretical part presents an overview of the global POS systems market, which reached a value of 17 trillion USD in 2024, and compares local and cloud-based solutions. It provides a detailed examination of the Slovenian legislative framework for fiscal receipt verification through the FURS system, analysis of security standards (ISO/IEC 27001:2022, ISO/IEC 27002:2022, GDPR, NIS2, ZInfV-1), and principles of modern user experience in business applications.

The practical part describes the comprehensive development of the application using .NET 8.0, Avalonia UI framework, Entity Framework Core, and Microsoft SQL Server technologies. Clean Architecture is implemented with clearly separated layers (presentation, business logic, data access), MVVM architectural pattern, and Repository Pattern for data access. Special attention is given to security mechanisms, including BCrypt password hashing, GDPR compliance in personal data minimization, secure HTTPS/TLS communication, and SQL injection prevention.

The application enables fast and intuitive use with a clearly structured user interface, user authentication with role-based access, product and inventory management, payment processing, and simulation of fiscal verification through the FURS sandbox environment. The developed solution represents a cost-effective alternative to expensive commercial systems, particularly for small businesses, sole proprietors, and hospitality establishments.

Keywords: *POS system, cash register application, FURS integration, information security, desktop application*

KAZALO VSEBINE

KAZALO SLIK

KAZALO TABEL

KAZALO GRAFIKONOV

1 UVOD

V sodobnem poslovnem okolju je učinkovita programska podpora ključna za uspešno delovanje podjetij. Blagajniško poslovanje mora biti hitro, zanesljivo in skladno z zakonodajo, vključno z obveznim davčnim potrjevanjem preko sistema FURS.

Kljub temu številna podjetja še vedno uporabljajo zapletene ali zastarele blagajniške rešitve, ki otežujejo delo uporabnikom in povečujejo možnost napak. Dodatni izziv predstavlja tudi zaščita občutljivih podatkov pred morebitnimi zlorabami.

Zato se to diplomsko delo osredotoča na razvoj preproste, varne in uporabniku prijazne namizne aplikacije za blagajniško poslovanje, ki vključuje povezavo s FURS. Delo združuje teoretični pregled področja in praktično implementacijo, s poudarkom na uporabniški izkušnji, zakonodajnih zahtevah ter osnovnih varnostnih mehanizmih.

1.1 Opis področja in opredelitev problema

Sodobno blagajniško poslovanje zahteva zanesljive in učinkovite programske rešitve, ki so preproste za uporabo in hkrati skladne z zakonodajo, kot je sistem davčnega potrjevanja FURS. V praksi sem kot študent in kot zaposlen opazil, da številna podjetja še vedno uporabljajo zastarele ali preveč zapletene aplikacije, ki otežujejo osnovna opravila. Pri delu sem se pogosto srečeval z nepreglednimi vmesniki, nejasnimi funkcijami in postopki, ki zahtevajo preveč korakov za preproste naloge. Posledično prihaja do napačne uporabe sistema in nezadovoljstva med uporabniki.

Motivacijo za obravnavo te tematike v diplomskem delu sem našel v želji, da razvijem nekaj resnično uporabnega, kar bo olajšalo delo končnim uporabnikom. Hkrati sem si zastavil izziv samostojnega razvoja kompleksne namizne aplikacije, ki vključuje tudi povezavo z zunanjimi sistemi. Takšna rešitev bi bila še posebej koristna za manjša podjetja, samostojne podjetnike, trgovine, franšize, servise in gostinske lokale, ki pogosto nimajo dostopa do dragih ali uporabniku prijaznih sistemov.

1.2 Namen, cilji in osnovne trditve

Namen diplomskega dela je razvoj sodobne, preproste in uporabniku prijazne namizne aplikacije za blagajniško poslovanje, ki vključuje povezavo s sistemom FURS. S tem želim prispevati k izboljšanju uporabniške izkušnje in pokazati, da je možno takšno rešitev razviti samostojno in učinkovito tudi z omejenimi viri. Diplomsko delo je priložnost, da znanje iz razvoja aplikacij uporabim na konkretnem primeru in se preizkusim v celoviti implementaciji poslovne rešitve.

Cilji dela so:

- razvita funkcionalna namizna aplikacija za blagajniško poslovanje,
- vključena podpora za davčno potrjevanje računov (FURS),
- uporabniški vmesnik, ki omogoča hitro in intuitivno uporabo,
- implementirani osnovni varnostni mehanizmi (npr. šifriranje, avtorizacija),
- izvedena analiza uporabniške izkušnje.

Na podlagi zastavljenega problema in ciljev sem oblikoval naslednje hipoteze:

- H1: Uporabniki pričakujejo hitro in intuitivno blagajniško rešitev, ki je preprosta za uporabo že ob prvi uporabi.
- H2: Zanesljiva integracija z davčnim potrjevanjem FURS je ključna za nemoteno delovanje blagajniškega sistema.
- H3: Uporaba lokalne namizne aplikacije predstavlja stroškovno učinkovitejšo rešitev za mala podjetja kot naročnina na oblačne rešitve POS.
- H4 (kibernetska varnost): Vključitev varnostnih mehanizmov, kot sta šifriranje in zaščita dostopa, pomembno zmanjša možnost zlorabe podatkov v aplikacijah POS.
- H5: Uporabniki bodo aplikacijo z dobrim uporabniškim vmesnikom ocenili višje glede na zadovoljstvo in zaupanje v uporabo.

1.3 Predpostavke in omejitve

Pričakujem, da se bom pri izvedbi diplomskega dela srečal z nekaterimi predpostavkami in omejitvami, kot so:

- celoten razvoj poteka samostojno, zato je razpoložljiv čas za izvedbo funkcionalnosti omejen;
- integracija s sistemom FURS bo izvedena preko okolja sandbox, saj produkcijski dostop ni na voljo;
- simulacija povezave POS z banko bo ponazorjena zgolj na funkcionalni ravni in ne vključuje dejanskega prenosa podatkov;
- število sodelujočih pri testiranju uporabniške izkušnje bo omejeno na manjšo skupino znancev in sodelavcev.

1.4 Uporabljene raziskovalne metode

Pri izdelavi diplomskega dela bom uporabil kombinacijo **teoretične in empirične raziskovalne metode**. Teoretični del bo vključeval pregled literature s področij blagajniških sistemov, zakonodajnega okvira FURS, varnostnih mehanizmov v poslovnih aplikacijah in uporabniške izkušnje (UI/UX).

V empiričnem delu bo poudarek na **razvoju in implementaciji aplikacije**, ki bo temeljila na metodi **analize in sinteze** – z analizo obstoječih rešitev, načrtovanjem lastne arhitekture in preizkušanjem funkcionalnosti. V fazi testiranja bo uporabljena tudi **metoda opazovanja** uporabnikov, ki bodo preizkusili aplikacijo in podali povratne informacije.

Pridobljene podatke in opažanja bom uporabil za **ovrednotenje hipotez** ter oblikovanje zaključkov in priporočil za nadaljnji razvoj rešitve.

2 TEORIJA

2.1 Pregled obstoječih POS/blagajniških rešitev

Namizne rešitve POS predstavljajo klasičen pristop k blagajniškemu poslovanju, pri katerem je programska oprema nameščena lokalno na računalniku uporabnika. Takšen arhitekturni model omogoča delovanje tudi brez stalne internetne povezave, kar podjetjem zagotavlja večjo avtonomijo in neodvisnost od zunanjih ponudnikov storitev. Raziskave s področja razvoja sistemov POS za rastoča gospodarstva poudarjajo, da lokalne implementacije prinašajo prednosti predvsem na področju nadzora nad občutljivimi podatki in zmanjšanja dolgoročnih stroškov delovanja, saj ni potrebe po mesečnih naročninah. Kljub tem prednostim pa lokalni sistemi zahtevajo več tehničnega znanja za vzdrževanje, nameščanje varnostnih posodobitev in upravljanje varnostnih kopij podatkov. (Sebastina Nkechi Okofu, 2025)

Oblačne rešitve POS predstavljajo sodobnejši in vse bolj razširjen pristop, pri katerem sta aplikacija in podatkovna baza gostovani v oblaku. Globalni trg oblačnih sistemov POS je leta 2022 znašal 3,9 milijarde USD, napovedi pa kažejo izjemno rast na več kot 30 milijard USD do leta 2031, s povprečno letno stopnjo rasti skoraj 25 %. (The Payments Association, 2024) Med glavnimi prednostmi oblačnih sistemov so dostopnost od kjerkoli, samodejne posodobitve programske opreme, integrirane varnostne kopije podatkov in nižji začetni stroški uvedbe. Slabosti tega modela pa so odvisnost od stalne internetne povezave in ponavljajoči se stroški naročnin. Analize stroškov lastništva kažejo, da je izbira med lokalnim in oblačnim sistemom odvisna predvsem od velikosti podjetja, tehnične usposobljenosti zaposlenih in dolgoročne finančne strategije. (Research, 2024)

Sodobni trendi razvoja sistemov POS vse bolj poudarjajo pomen uporabniške izkušnje kot enega ključnih dejavnikov uspešnosti. Raziskave na področju personalizacije blagajniških vmesnikov kažejo, da analiza vedenja uporabnikov in prilagajanje vmesnika glede na zbrane podatke pomembno prispevata k večjemu zadovoljstvu strank ter učinkovitosti prodajnega procesa. To postaja še posebej pomembno v času vse večje digitalizacije trgovine, ko uporabniki pričakujejo intuitivne vmesnike, ki omogočajo hitro uporabo brez obsežnega usposabljanja. (OKUMOKU-EVRORO, 2025)

2.2 Zakonodajni okvir za davčno potrjevanje računov (FURS)

V skladu z zakonodajo Republike Slovenije morajo zavezanci za DDV ob izdaji računov uporabljati sistem za davčno potrjevanje računov, ki ga upravlja Finančna uprava Republike Slovenije (FURS). Ta postopek omogoča elektronsko preverjanje avtentičnosti izdanih računov v realnem času ter preprečuje zlorabe in prikrivanje prihodkov.

Sistem temelji na **elektronskem podpisovanju in pošiljanju podatkov preko posebnega spletnega vmesnika (angl. *Application Programming Interface*, API)**, ki deluje bodisi v produkcijskem okolju bodisi v **(testnem) okolju sandbox**. Slednje omogoča razvijalcem programske opreme preverjanje delovanja integracije brez vpliva na dejanske evidence. Davčno potrjeni računi vsebujejo posebne identifikatorje, kot sta **EOR (enolična oznaka računa)** in **ZOI (zaščitna oznaka izdajatelja)**.

V okviru diplomskega dela bo integracija z davčno blagajno izvedena simulacijsko, preko **okolja FURS sandbox**, saj produkcijska vključitev zahteva ustrezno registracijo programske opreme in certifikate. Tehnična dokumentacija, objavljena s strani FURS, je javno dostopna in služi kot osnova za razumevanje zahtev, strukture podatkov in pravil pri vzpostavitvi komunikacije med aplikacijo in sistemom FURS. (FURS, 26. 8. 2025)

2.3 Varnostni vidiki v poslovnih aplikacijah

Informacijska varnost predstavlja temeljni vidik razvoja sodobnih poslovnih aplikacij, še posebej tistih, ki obdelujejo finančne transakcije in osebne podatke uporabnikov. Mednarodni standard ISO/IEC 27001:2022 definira zahteve za vzpostavitev, implementacijo, vzdrževanje in stalno izboljševanje sistema upravljanja informacijske varnosti (ISMS) v organizacijah (ISO, 2022). Standard temelji na holističnem pristopu k informacijski varnosti, ki vključuje ljudi, procese in tehnologijo, s čimer predstavlja orodje za obvladovanje tveganj, kibernetsko odpornost in operativno odličnost (ISO, 2022). Za razvoj aplikacij POS je ključnega pomena razumevanje načel varovanja podatkov, vključno z zaščito pred nepooblaščenim dostopom, integriteto podatkov in razpoložljivostjo storitev.

Medtem ko ISO/IEC 27001 določa zahteve za ISMS, standard ISO/IEC 27002:2022 ponuja podrobne smernice in najboljše prakse za implementacijo varnostnih kontrol. Nova revizija iz leta 2022 je prinesla pomembne spremembe v strukturi standarda, saj je 114 kontrol iz prejšnje verzije zmanjšala na 93, hkrati pa jih reorganizirala iz 14 domen v 4 glavne kategorije:

organizacijske, tehnološke, fizične in človeške kontrole. (UpGuard, 2024) Vsaka kontrola v novi verziji vključuje tabelo atributov in jasen cilj implementacije, kar omogoča organizacijam boljše razumevanje, kdaj in kako uporabiti posamezno kontrolo. (ISACA, 2023)

V okviru razvite aplikacije bodo vključeni naslednji varnostni elementi:

- **avtorizacija in dostop**: uporabniški dostop bo omejen preko prijave, kjer bodo določene funkcije dostopne le določenim vlogam (npr. administratorjem);
- **šifriranje občutljivih podatkov**: gesla in drugi občutljivi lokalni podatki bodo ustrezno šifrirani;
- **varna povezava (HTTPS/TLS)**: za prenos podatkov med aplikacijo in zunanjimi storitvami, kot sta FURS (sandbox) in simulirani bančni vmesnik POS, bo uporabljena zaščitena povezava preko HTTPS, kot zahteva zakonodaja;
- **možna vključitev dvofaktorske avtentikacije (2FA)**: glede na razpoložljiv čas in kompleksnost implementacije se bo razmislilo o uvedbi dodatne zaščite ob prijavi.

Vse varnostne rešitve bodo izvedene v okviru razvojnega okolja in niso namenjene za produkcijsko uporabo, temveč kot funkcionalna simulacija, ki sledi realnim zahtevam v varnostnem smislu.

2.4 Uporabniška izkušnja in sodoben UI/UX

Raziskave uporabniških pričakovanj v sistemih POS kažejo, da 45 % strank pričakuje personalizirane nakupovalne izkušnje v fizični trgovini, ki se ujemajo z njihovimi spletnimi izkušnjami, kar spodbuja povpraševanje po inteligentnih rešitvah POS. To postavlja uporabniško izkušnjo (UX) in uporabniški vmesnik (UI) v središče razvoja sodobnih poslovnih aplikacij. (The Payments Association, 2024)

Uporabniška izkušnja (UX – *user experience*) in uporabniški vmesnik (UI – *user interface*) sta ključna elementa pri razvoju sodobne programske opreme. Cilj je ustvariti intuitiven, učinkovit in prijeten način interakcije med človekom in sistemom.

Dober uporabniški vmesnik omogoča:

- hitro učenje uporabe programa,
- zmanjšanje števila napak,

- večje zadovoljstvo uporabnikov,
- boljšo učinkovitost pri vsakodnevnih opravilih.

V teoriji obstajajo različna načela za oblikovanje uporabniških vmesnikov, kot na primer:

- **jasnost**: uporabnik mora razumeti, kaj program počne;
- **konsistentnost**: podobne funkcije naj delujejo na enak način;
- **vidnost**: pomembne informacije naj bodo vidne brez dodatnega iskanja;
- **povratne informacije**: sistem naj jasno pokaže, kaj se dogaja (npr. uspešno shranjevanje);
- **toleranca na napake**: omogočeni naj bodo razveljavitev dejanj, opozorila pri napačnih vnosih ipd.

V kontekstu poslovnih aplikacij so še posebej pomembne:

- **prilagojenost vsakodnevnim opravilom**,
- **preprostost klikanja/navigacije**,
- **dostopnost tudi manj izkušenim uporabnikom**.

Zasnova UI/UX se pogosto povezuje z orodji za oblikovanje (npr. Figma, Adobe XD), pa tudi s tehnologijami, ki omogočajo odzivne vmesnike (npr. .NET WPF, HTML5/CSS3 v spletnih okoljih).

2.5 Primerjava lokalnih in oblačnih rešitev

Pri izbiri programske rešitve za blagajniško poslovanje se podjetja pogosto odločajo med **lokalnimi (on-premise)** in **oblačnimi (cloud-based)** aplikacijami. Obe možnosti imata svoje prednosti in slabosti, izbira pa je odvisna od potreb, velikosti podjetja, varnostnih zahtev in stroškovne učinkovitosti.

Lokalne rešitve so nameščene neposredno na računalnik uporabnika ali strežnik znotraj podjetja. Takšne rešitve običajno ne zahtevajo stalne internetne povezave in omogočajo popoln nadzor nad podatki. V kontekstu manjših podjetij je to pogosto prednost, saj omogoča večjo **neodvisnost od zunanjih ponudnikov** in **nižje dolgoročne stroške**, ker ni naročnin.

Slabosti lokalnih rešitev vključujejo večjo **odgovornost za varnost, posodabljanje in vzdrževanje**, kar je lahko za manjša podjetja z omejenimi tehničnimi znanji zahtevno.

Oblačne rešitve, kot so iPOS, eBlagajna ali Birokrat POS, omogočajo uporabo sistema preko spletnega vmesnika ali mobilnih aplikacij. Glavna prednost je **preprost dostop od kjerkoli**, brez potrebe po lokalni namestitvi. Oblačni sistemi omogočajo avtomatske nadgradnje, **backup podatkov** in pogosto vključujejo **tehnično podporo** v okviru naročnine.

Vendar pa so ti sistemi praviloma vezani na **mesečne ali letne naročnine**, kar lahko dolgoročno predstavlja višje stroške. Prav tako se pojavlja skrb glede **zasebnosti in varnosti podatkov**, saj so le-ti shranjeni na strežnikih zunanjega ponudnika.

Tabela 1: Primerjava lokalnih in oblačnih rešitev POS

Lastnost	**Lokalna rešitev**	**Oblačna rešitev**
Dostopnost	Omejena na eno lokacijo	Dostopno kjerkoli
Stroški	Enkratni ali lastni razvoj	Mesečna/letna naročnina
Vzdrževanje	Uporabnik sam/oddelek IT	Ponudnik storitve
Varnost podatkov	V rokah uporabnika	Odvisna od ponudnika
Internetna povezava	Ni potrebna/razen POS in FURS	Obvezna
Nadgradnje	Ročne	Avtomatske
Prilagodljivost	Popolna kontrola	Omejena

(Lastni vir, 2025)

2.6 Standardi ISO in kibernetska varnost

2.6.1 Mednarodni standardi informacijske varnosti

V kontekstu naraščajočih kibernetskih groženj in vse večje digitalizacije poslovanja so mednarodni standardi informacijske varnosti postali ključno orodje za zagotavljanje ustrezne ravni zaščite informacijskih sistemov. Organizacije, ki razvijajo ali implementirajo sisteme POS, morajo upoštevati vrsto standardov in regulativnih zahtev.

2.6.2 ISO/IEC 27001 in 27002

Standard ISO/IEC 27001:2022 predstavlja globalno najbolj priznan standard za sisteme upravljanja informacijske varnosti z več kot 70.000 izdanimi certifikati v 150 državah po podatkih iz leta 2022. (ISO, 2022) Standard omogoča organizacijam, da postanejo bolj ozaveščene o tveganjih in proaktivno identificirajo in naslovijo šibke točke v svojih sistemih. Revizija iz oktobra 2022 je prinesla posodobljene zahteve in revidiran nabor kontrol v prilogi A, kar odraža razvoj kibernetskih groženj in novih tehnologij. (Secureframe, 2024)

Dopolnilni standard ISO/IEC 27002:2022, objavljen februarja 2022, zagotavlja referenčni nabor generičnih kontrol informacijske varnosti, vključno s smernicami za implementacijo. (ISO, 2022) Nova verzija je uvedla 11 novih kontrol, med njimi napovedovanje groženj (threat intelligence), varnost v oblaku, pripravljenost IKT za poslovno kontinuiteto in varovanje fizičnih prostorov med incidenti. (ISACA, 2023) Organizacije, ki so certificirane po ISO 27001:2013, morajo zaključiti prehod na novo verzijo do 31. oktobra 2025. (Secureframe, 2024)

2.6.3 Splošna uredba o varstvu podatkov (GDPR)

Splošna uredba o varstvu podatkov (GDPR), ki je začela veljati 25. maja 2018, predstavlja enega najstrožjih zakonov o zasebnosti podatkov na svetu. (European Union Agency for Fundamental Rights, 2024) Uredba se uporablja za organizacije s sedežem v EU in za vse entitete, ki obdelujejo osebne podatke državljanov EU, ne glede na njihovo geografsko lokacijo. GDPR zahteva več ključnih načel obdelave podatkov: zakonitost in preglednost, omejitev namena, minimizacija podatkov, točnost, omejitev hrambe ter odgovornost. (Crown Information Management, 2024)

Po šestih letih izvajanja GDPR je Evropska komisija leta 2024 predlagala več sprememb za poenostavitev postopkov in izboljšanje čezmejnega sodelovanja med nadzornimi organi. (Crown Information Management, 2024) Posebna pozornost je namenjena tudi varstvu otrok v

digitalnem okolju, saj so regulatorji povečali izvajanje ukrepov za zaščito mladoletnikov na spletu. (Covington & Burling, 2024) Za sisteme POS, ki obdelujejo osebne podatke strank, je skladnost z GDPR obvezna, kar vključuje implementacijo tehničnih in organizacijskih ukrepov za zaščito podatkov.

2.6.4 Direktiva NIS2 in kibernetska varnost v EU

Direktiva o ukrepih za visoko skupno raven kibernetske varnosti v Uniji (NIS2), sprejeta decembra 2022, predstavlja pomembno posodobitev prvotne direktive NIS iz leta 2016. (European Commission, 2022) Države članice so morale novo direktivo prenesti v svojo nacionalno zakonodajo do 17. oktobra 2024, vendar je ta rok večina zamudila. (Skadden, 2024)

NIS2 bistveno širi obseg zavezancev v primerjavi s prejšnjo direktivo in uvaja strožje zahteve glede obvladovanja kibernetskih tveganj ter poročanja o varnostnih incidentih. (Skadden, 2024) Organizacije morajo prvo obvestilo o pomembnem incidentu podati v 24 urah po zaznavi, v 72 urah pa oddati podrobno poročilo. Končno poročilo morajo predložiti najpozneje v enem mesecu. (ECSO, 2024)

Kazni za neskladnost z določbami so lahko zelo visoke – za ključne subjekte dosegajo tudi do 10 milijonov evrov oziroma 2 % letnega globalnega prometa. (Wavestone, 2024) Ena izmed večjih novosti direktive je tudi neposredna odgovornost upravnih organov za skladnost z zahtevami NIS2. To pomeni, da lahko pristojni organi v primeru ponavljajočih se kršitev celo razrešijo člane vodstva. (Skadden, 2024)

2.6.5 Slovenski zakon o informacijski varnosti (ZInfV-1)

Republika Slovenija je maja 2025 sprejela nov Zakon o informacijski varnosti (ZInfV-1), s katerim je v slovenski pravni red prenesla direktivo NIS2. (varnost, 2025) Zakon je začel veljati 19. junija 2025, čeprav je Slovenija zamudila rok iz direktive za sedem mesecev. (Uradni list, 2025) ZInfV-1 ureja področje informacijske in kibernetske varnosti ter določa nacionalni sistem informacijske varnosti v Sloveniji, pri čemer se ne uporablja za sisteme, ki varujejo tajne podatke. (varnost, 2025)

Nova zakonodaja prinaša bistveno širši obseg zavezancev v primerjavi s prejšnjim ZInfV, saj se zavezanci delijo na bistvene in pomembne subjekte glede na sektor delovanja ter velikost organizacije. (SI-CERT, 2024) Podjetja, ki imajo več kot 50 zaposlenih in letni promet nad 10 milijonov EUR ter delujejo v kritičnih sektorjih, morajo izpolnjevati zahteve zakona. Zakon določa tudi obsežne naloge za nacionalno skupino CSIRT SI-CERT, vključno z vzpostavitvijo

sistema za proaktivni pregled omrežnih sistemov in koordinacijo usklajenega razkrivanja ranljivosti. (SI-CERT, 2024)

2.6.6 TISAX za avtomobilsko industrijo

TISAX (Trusted Information Security Assessment Exchange) je specializiran standard informacijske varnosti za avtomobilsko industrijo, ki ga upravlja združenje ENX. (DQS, 2024) Temelji na katalogu zahtev VDA ISA (Information Security Assessment) nemškega združenja avtomobilske industrije in je tesno usklajen z mednarodnim standardom ISO/IEC 27001, vendar vključuje tudi dodatne zahteve, prilagojene specifičnim potrebam avtomobilskega sektorja. (TÜV Rheinland, 2024)

Oktobra 2023 je bila objavljena nova verzija kataloga ISA 6.0, ki je obvezna za vse ocene TISAX, naročene po 1. aprilu 2024. (DEKRA, 2024) Nova različica uvaja okrepljene kontrole za zaščito pred napadi ransomware in izboljšuje odpornost proti napredno vztrajnim grožnjam (APT). Prav tako dodatno poudarja pomen varnosti informacijske tehnologije (IT) in operacijske tehnologije (OT). (ISDDecisions, 2024)

3 PRAKTIČNI DEL – RAZVOJ APLIKACIJE

3.1 Izbira tehnologij in razvojnega okolja

Pri izbiri tehnologij za razvoj namizne aplikacije sem se moral odločiti med več možnostmi. Glavni kriteriji pri izbiri so bili:

- **cross-platform podpora** (možnost delovanja na različnih operacijskih sistemih),
- **moderna arhitektura in dolgoročna podpora,**
- **možnost integracije z zunanjimi sistemi (FURS API),**
- **razpoložljivost dokumentacije in skupnosti razvijalcev,**
- **lastno predznanje** in učna krivulja.

Po temeljiti primerjavi sem se odločil za **Avalonia UI Framework**, ki temelji na **platformi .NET 8.0**. Avalonia je odprtokodno razvojno okolje za razvoj cross-platform namiznih aplikacij, ki omogoča uporabo jezikov **XAML** za definicijo uporabniškega vmesnika in **C#** za poslovno logiko. (Avalonia, 2025) Ta tehnologija omogoča razvoj aplikacije, ki lahko teče na operacijskih sistemih **Windows, Linux in macOS**, kar predstavlja pomembno prednost v primerjavi z omejeno tehnologijo Windows Presentation Foundation (WPF).

Za razvojno okolje sem izbral **Visual Studio 2022**, ki ponuja odlično podporo za razvoj .NET, inteligenten code completion (IntelliSense), integriran debugger in obsežna orodja za upravljanje paketov preko sistema NuGet.

Za upravljanje podatkovne baze sem se odločil za **Entity Framework Core 8.0** kot orodje ORM (Object-Relational Mapping), ki poenostavi delo s podatkovno bazo, in **Microsoft SQL Server** kot sistem za upravljanje relacijskih baz podatkov.

3.2 Arhitekturni pristop

Pri načrtovanju arhitekture aplikacije sem sledil načelom **Clean Architecture** in principom **SOLID**, kar omogoča lažje vzdrževanje, testiranje in razširljivost sistema.

Aplikacijo sem razdelil na več samostojnih plasti.

- **Predstavitvena plast (Presentation Layer)**: ta plast vsebuje uporabniški vmesnik, zgrajen z Avalonia UI Framework. Vključuje datoteke XAML za definicijo videza in ViewModele, ki implementirajo arhitekturni vzorec **MVVM (Model-View-ViewModel)**. Vzorec MVVM omogoča jasno ločevanje logike prikaza od vizualnih elementov in olajša testiranje in vzdrževanje kode.
- **Poslovna logika (Business Logic Layer)**: srednja plast vsebuje poslovne storitve (Services), ki implementirajo celotno poslovno logiko aplikacije. Tukaj potekajo obdelava podatkov, validacije, izračuni DDV-ja, generiranje številk računov in orkestracija med različnimi komponentami sistema. Prav tako so v tej plasti **DTOs (Data Transfer Objects)**, ki služijo prenosu podatkov med plastmi brez neposrednega izpostavljanja entitet podatkovne baze.
- **Podatkovna plast (Data Access Layer)**: najnižja plast implementira **Repository Pattern** za dostop do podatkov. Vzorec Repository omogoča abstrakcijo podatkovne baze in centralizacijo vseh operacij branja in pisanja. To omogoča preprosto testiranje z uporabo objektov mock in potencialno zamenjavo podatkovne baze brez večjih sprememb v višjih plasteh.
- Takšna arhitektura omogoča vzorec **Dependency Injection (DI)**, kjer so odvisnosti med komponentami injektirane preko konstruktorjev, kar omogoča **loose coupling** in boljšo testabilnost celotnega sistema.

3.3 Podatkovni model

Podatkovni model sem oblikoval tako, da odraža realne poslovne potrebe blagajniškega sistema. Osnova podatkovnega modela so štiri glavne entitete.

Entiteta User (Uporabnik): predstavlja uporabnike sistema (blagajnike, administratorje). Vsebuje atribute, kot so uporabniško ime, šifrirano geslo, polno ime, vloga (Admin/Cashier) in status aktivnosti. Posebno pozornost sem namenil shranjevanju gesel, kjer sem implementiral **algoritem BCrypt hashing** z vgrajenim mehanizmom salt, kar zagotavlja visoko stopnjo varnosti. Gesla niso nikoli shranjena v berljivi obliki (plain text), kar je ključnega pomena za varnost sistema.

Entiteta Product (Izdelek): vsebuje vse podatke o izdelkih v sistemu – edinstveno kodo, naziv, ceno, stopnjo DDV-ja, količino na zalogi in status aktivnosti. Posebej sem se posvetil pravilni definiciji podatkovnih tipov, kjer sem za monetarne vrednosti uporabil **decimalna števila**, ki zagotavljajo natančnost pri finančnih izračunih brez napak zaokroževanja, ki so značilne za tipe float ali double.

Entiteta Receipt (Račun): predstavlja izdani račun in vsebuje vse potrebne podatke za davčno potrjevanje. Poleg osnovnih podatkov (številka računa, datum in čas izdaje, način plačila, skupni znesek) vsebuje tudi **ZOI (Zaščitna Oznaka Izdajatelja)** in **EOR (Enkratna Oznaka Računa)**, ki sta ključna elementa sistema FURS. Pomembna je tudi informacija o ID-ju blagajnika, kjer sem se odločil shranjevati **samo ID namesto polnega imena**, kar je v skladu z **uredbo GDPR** o minimizaciji osebnih podatkov. Status računa sem implementiral kot enum z vrednostmi Pending, Completed, Cancelled in FursError, kar omogoča sledenje življenjskega cikla računa.

Entiteta ReceiptItem (Postavka računa): vsebuje posamezne artikle na računu. Pomembno je, da sem implementiral **pristop snapshot** – to pomeni, da se podatki o izdelku (naziv, cena, DDV) shranijo v trenutku prodaje in se ne spreminjajo, tudi če se kasneje spremenijo podatki v tabeli Products. To je ključno za ohranjanje integritete računov in skladnosti z davčno zakonodajo.

Med entitetami sem vzpostavil **relacije**, predvsem **one-to-many** med Receipt in ReceiptItem, kar omogoča, da ima vsak račun več postavk. Implementiral sem tudi **cascade delete**, kar pomeni, da ob izbrisu računa avtomatsko izbrišemo vse pripadajoče postavke.

3.4 Entity Framework Core in migracije

Za komunikacijo s podatkovno bazo sem uporabil **Entity Framework Core**, ki omogoča objektno-relacijsko preslikavo (ORM). To pomeni, da lahko z bazo komuniciram preko objektov C# in poizvedb LINQ, namesto pisanja neposrednih stavkov SQL.

Kreiral sem razred **AppDbContext**, ki predstavlja sejo za dostop do baze. V tem razredu sem definiral lastnosti **DbSet** za vsako entiteto in konfiguriral metodo **OnModelCreating**, kjer sem natančno določil:

- **primarne ključe** za vse tabele,

- **unique indexe** na poljih, kot so Username, ProductCode in ReceiptNumber, kar preprečuje podvojene vnose,
- obvezna polja, ki ne smejo biti NULL,
- omejitve **MaxLength** za tekstovna polja,
- **Decimal precision** (18,2) za monetarne vrednosti,
- povezave **Foreign Key** in **pravila cascade**.

Povezavo do podatkovne baze sem definiral v datoteki **appsettings.json**, kar omogoča preprosto spreminjanje nastavitev brez potrebe po rekompilaciji kode. Za kreiranje podatkovne baze sem uporabil **pristop Code-First** z migracijami Entity Framework. Pristop Code-First pomeni, da se API (ali druga programska rešitev) začne razvijati neposredno s pisanjem kode – brez predhodnega podrobnega načrta API-ja v obliki specifikacije (npr. Swagger). Dokumentacija in specifikacije API-ja so tako ustvarjene šele naknadno, na podlagi že obstoječe kode. Ta pristop je primeren, ko je hitra dostava ključna in gre pogosto za manjše ali enkratne projekte, kjer ni potrebe po obsežni ponovni uporabi ali sodelovanju med več ekipami. V Package Manager Console sem izvršil ukaz Add-Migration InitialCreate, ki je avtomatsko generiral kodo SQL za kreiranje tabel na podlagi mojih entitet. Z ukazom Update-Database sem nato apliciral migracijo in kreiral fizično bazo podatkov. Prednost takšnega pristopa je **verzioniranje sheme baze** – vsaka sprememba modela se zabeleži kot nova migracija, kar omogoča preprosto sledenje spremembam in možnost vrnitve na prejšnjo verzijo, če je treba.

3.5 Implementacija Repository Patterna

Za abstrakcijo dostopa do podatkov sem implementiral **Repository Pattern**. Ta vzorec omogoča ločitev podatkovne logike od poslovne logike in centralizacijo operacij z bazo.

Začel sem z vmesnikom **IGenericRepository<T>**, ki definira standardne operacije CRUD (Create, Read, Update, Delete):

- GetByIdAsync(int id) – pridobi zapis po ID-ju,
- GetAllAsync() – pridobi vse zapise,

- FindAsync(Expression<Func<T, bool>> predicate) – omogoča iskanje po poljubnih pogojih,
- AddAsync(T entity) – doda nov zapis,
- UpdateAsync(T entity) – posodobi obstoječ zapis,
- DeleteAsync(T entity) – izbriše zapis.

Vse metode sem implementiral kot **asinhrone (async)**, kar preprečuje blokiranje uporabniškega vmesnika med operacijami z bazo in zagotavlja boljšo odzivnost aplikacije.

Implementiral sem tudi **specifične repozitorije** za vsako entiteto, kjer sem dodal dodatne metode, specifične za posamezno entiteto:

- **UserRepository** vsebuje metodo GetByUsernameAsync, ki je ključna za avtentikacijo uporabnikov,
- **ProductRepository** ima metodo GetByCodeAsync za iskanje izdelkov po črtni kodi,
- **ReceiptRepository** vključuje metodi GetByReceiptNumberAsync in

  GetReceiptsByCashierAsync, pri čemer uporablja **eager loading** s pomočjo metode .Include(), da v enem klicu pridobi tudi vse postavke računa.

3.6 Poslovna logika – Storitve (Services)

Storitve predstavljajo srce poslovne logike aplikacije. Vsaka storitev je odgovorna za specifično poslovno domeno.

UserService: implementira avtentikacijo uporabnikov. Metoda AuthenticateAsync sprejme uporabniško ime in geslo, pridobi uporabnika iz baze preko repositoryja ter preveri geslo z metodo **BCrypt.Verify**. Ta metoda primerja vneseno geslo s shranjeno šifro in vrne pravilno, če se ujemata. Če je avtentikacija uspešna, vrne uporabnika, sicer ne vrne nič. To omogoča preprosto preverjanje v uporabniškem vmesniku.

ProductService: upravlja z izdelki in uporablja vzorec **DTO (Data Transfer Objects)**. Vsi podatki, ki jih vrne storitev, so v obliki objektov ProductDto, kar preprečuje neposredno

izpostavljanje podatkovnih entitet v predstavitveni plasti. Storitev vsebuje tudi validacijsko logiko, na primer preverjanje edinstvene kode izdelka pred dodajanjem novega vnosa.

ReceiptService: to je najbolj kompleksna storitev, ki upravlja z računi. Pri kreiranju novega računa izvede več korakov:

1. generira edinstveno **številko računa** v formatu R + datum + zaporedna številka (npr. R20250110-0001),
2. nastavi trenutni **datum in čas izdaje**,
3. kreira objekte **ReceiptItem** iz vseh izdelkov v košarici,
4. shrani račun v bazo, kjer EF Core avtomatsko generira ID,
5. vrne kreirani račun z vsemi podatki.

Metoda GetReceiptByIdAsync uporablja predhodno nalaganje za pridobitev vseh povezanih postavk v enem stavku SQL, kar je bistveno bolj učinkovito kot več ločenih poizvedb.

FursService: to je najkompleksnejša storitev, zadolžena za komunikacijo s sistemom FURS. Ključna metoda ConfirmReceiptAsync izvede naslednje korake:

1. **generiranje ZOI (Zaščitna Oznaka Izdajatelja)**: ZOI je šifra MD5 specifičnega niza podatkov računa, ki vključuje davčno številko, datum in čas, številko računa, oznako poslovnega prostora, oznako blagajne in skupni znesek. Ta oznaka služi kot **zaščita pred manipulacijo** s podatki računa;
2. **kreiranje sporočila JSON**: sestavi strukturirano sporočilo v obliki JSON po specifikaciji FURS, ki vključuje vse potrebne podatke – davčno številko, identifikatorje računa, zneske, razčlenitev DDV in ZOI;
3. **simulacija sandbox**: ker v razvojni fazi nimam dostopa do produkcijskega certifikata FURS, sem implementiral **sandbox mode**, kjer se ZOI pravilno generira, medtem ko se EOR (Enkratna Oznaka Računa) simulira z naključnim GUID-om. Status računa se nastavi na "Completed".
4. **logging**: implementiral sem konzolno beleženje dogodkov za namen razhroščevanja in sledenja postopku potrjevanja.

Za **produkcijsko implementacijo** bi bilo treba:

- pridobiti digitalni certifikat od FURS,
- implementirati digitalno podpisovanje RSA-SHA256,
- vzpostaviti povezavo TLS 1.2/1.3,
- poslati zahtevo POST na FURS-ov API,
- procesirati odgovor in izvleči EOR.

3.7 Konfiguracija Dependency Injection

Za upravljanje odvisnosti med komponentami sem implementiral vzorec **Dependency Injection (DI)** z uporabo knjižnice Microsoft.Extensions.DependencyInjection.

Kreiral sem razred **ServiceCollectionExtensions** z metodo AddCoreServices, ki centralizirano registrira vse potrebne storitve:

- **DbContext** kot **Scoped** – nova instanca za vsako zahtevo/obseg,
- **Repositoryji** kot **Scoped** – ena instanca na obseg,
- storitve kot **Scoped** – zagotavlja, da dobijo vse storitve iste instance repozitorijev v istem obsegu,
- klient za komunikacijo FURS,
- **IConfiguration** kot **Singleton** – ena instanca za celotno aplikacijo.

V datoteki App.axaml.cs sem v metodi OnFrameworkInitializationCompleted implementiral:

1. **konfiguracijo**: naložil sem datoteko appsettings.json s ConfigurationBuilder,
2. **registracijo storitev**: kreiral ServiceCollection in registriral vse storitve,
3. **ServiceProvider**: zgradil ponudnika za reševanje odvisnosti,
4. **database seeding**: inicializiral testne podatke ob prvem zagonu.

3.8 Razred DatabaseSeeder – kreiranje testnih podatkov

Za olajšanje testiranja in razvoja sem implementiral razred **DatabaseSeeder**, ki avtomatsko kreira začetne podatke.

Metoda SeedUsersAsync: preveri, ali že obstaja uporabnik z administratorskimi pravicami in ga kreira, če ta še ne obstaja, z naslednjimi podatki:

- uporabniško ime: "admin",
- geslo: "admin123" (šifriran z BCrypt!),
- vloga: "Admin",
- polno ime: "Administrator".

Metoda SeedProductsAsync: če baza ne vsebuje izdelkov, doda 8 testnih izdelkov z različnimi stopnjami DDV:

- **živila (DDV 9,5 %)**: kruh, mleko, jogurt, maslo, konzervirana tunina,
- **pijače in ostalo (DDV 22 %)**: Coca Cola, čokolada, kava.

Inicializator podatkov se izvede avtomatsko ob zagonu aplikacije, kar omogoča takojšnje testiranje funkcionalnosti brez potrebe po ročnem vnašanju podatkov.

3.9 Arhitekturni vzorec MVVM

Pri razvoju uporabniškega vmesnika sem sledil arhitekturnemu vzorcu **MVVM (Model-View-ViewModel)**, ki je standard pri razvoju aplikacij z ogrodjem Avalonia in WPF. Vzorec MVVM jasno loči naslednje komponente:

- **Model**: predstavlja podatke in poslovno logiko. V mojem primeru so to entitete (User, Product, Receipt) in DTO-ji, ki jih uporabljam za prenos podatkov med plastmi;
- **View**: to so datotekeXAML, ki definirajo videz aplikacije. Komponenta View vsebuje samo označevalno kodo in ne vsebuje poslovne logike. Komunikacija s komponento ViewModel poteka izključno preko povezovanja podatkov;

- **ViewModel**: vmesna plast med View in Model, ki vsebuje logiko prikaza, ukaze (Commands) in lastnosti (Properties), na katere se veže View. ViewModel ne pozna Viewa, kar omogoča testiranje brez uporabniškega vmesnika;
- **implementacija ViewModelBase**: kreiral sem bazni razred ViewModelBase, ki implementira vmesnik **INotifyPropertyChanged**. Ta vmesnik je ključen za avtomatsko posodabljanje uporabniškega vmesnika ob spremembi podatkov. Implementiral sem pomožno metodo SetProperty<T>, ki:
  1. preveri, ali se je vrednost res spremenila,
  2. nastavi novo vrednost,
  3. sproži dogodek **PropertyChanged**, ki obvesti komponento View o spremembi,
  4. komponenta View se avtomatsko posodobi preko povezovanja podatkov,

- **implementacija RelayCommand**: za izvajanje akcij ob kliku na gumbe sem implementiral razred RelayCommand, ki implementira vmesnik **ICommand**. RelayCommand omogoča:
  - vezavo na metode ViewModela s pomočjo vezave podatkov,
  - podporo za generične parametre RelayCommand<T>,
  - implementacijo logike CanExecute, ki določa, kdaj je ukaz aktiven.

3.10 Komponenta Login Window – avtentikacija uporabnikov

Prvi vmesnik, s katerim se uporabnik sreča, je **prijavno okno (LoginWindow)**.

LoginViewModel vsebuje naslednje lastnosti:

- Username – vezava na tekstovno polje za uporabniško ime,
- Password – vezava na polje za geslo,
- ErrorMessage – prikaz morebitnih napak (npr. "Napačno uporabniško ime ali geslo"),
- HasError – indikator, ki določa vidnost sporočila tipa napaka,
- IsLoading – indikator nalaganja med avtentikacijo.

LoginCommand sproži metodo LoginAsync, ki izvede naslednje korake:

1. validacija vnosov: preveri, ali sta polji Username in Password izpolnjeni. Če ne, prikaže ustrezno napako;
2. klic UserService: asinhrono pokliče metodo AuthenticateAsync, ki preveri pravilnost prijave;
3. obdelava rezultata:

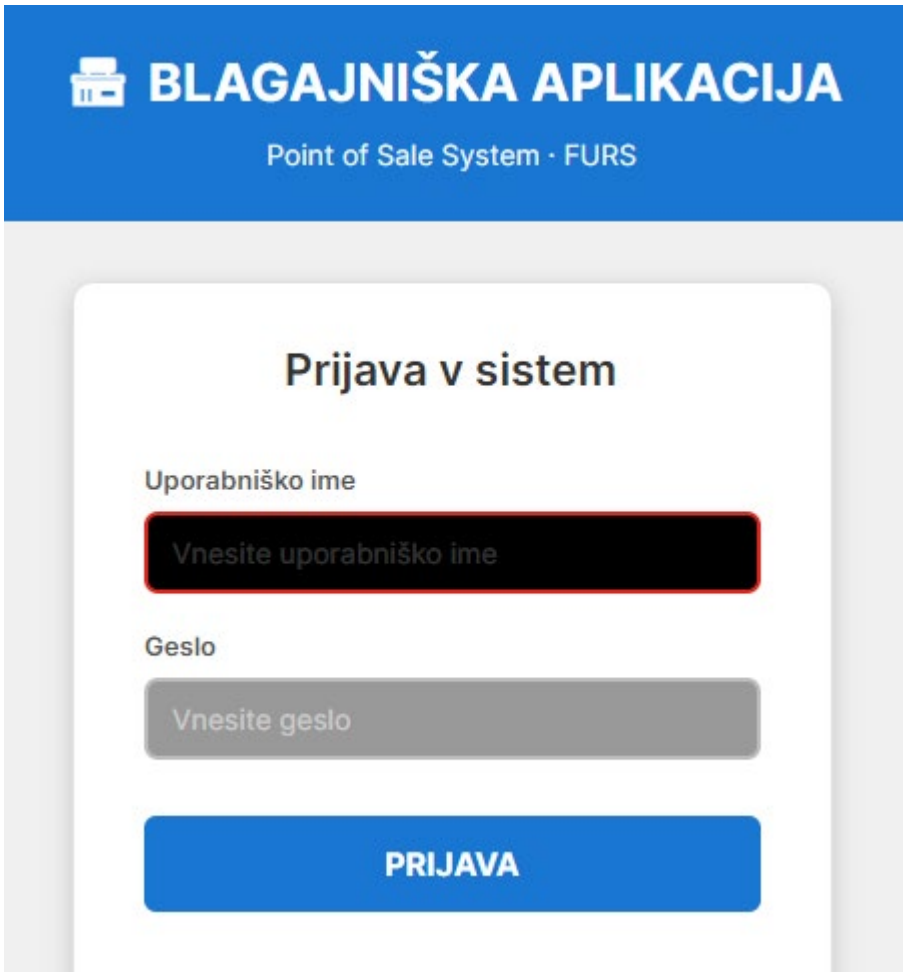
- če je prijava uspešna, shrani trenutnega uporabnika v statično lastnost App.CurrentUser, kreira novo instanco MainWindow, jo prikaže in zapre LoginWindow,
- če prijava ni uspešna, prikaže napako "Napačno uporabniško ime ali geslo";

4. ravnanje z napakami: v primeru izjeme (npr. težave s povezavo na bazo) ujame napako in prikaže uporabniku jasno sporočilo napake.

LoginWindow.axaml – vizualna oblika: prijavno okno sem oblikoval z naslednjimi elementi:

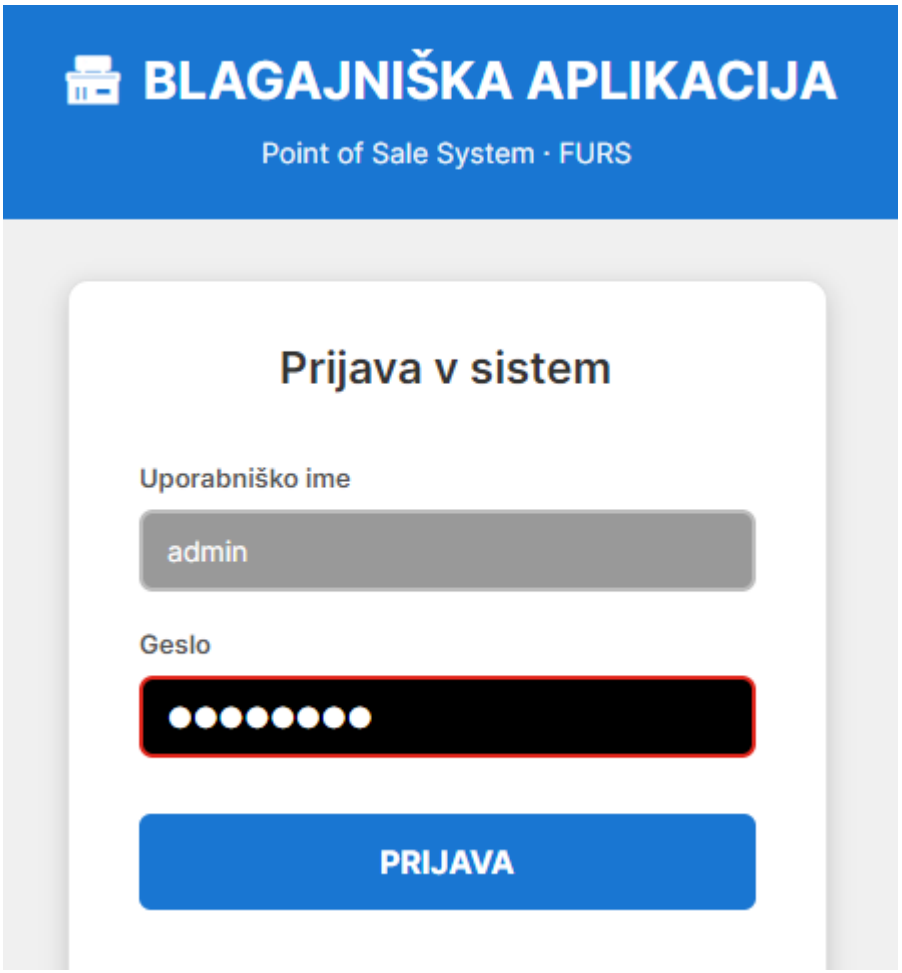
- **gradientno ozadje** (modre barve) za profesionalen videz,
- **centrirana kartična oblika** za 3D učinek,
- **logo in naslov** aplikacije na vrhu,
- **ikone** za boljšo vizualno komunikacijo,
- besedilno polje za **Username** z besedilom "Uporabniško ime",
- besedilo polje za **Password** z atributom PasswordChar, ki skrije znake gesla,
- besedilno polje za napake v rdeči barvi, ki se prikaže le, ko se parameter HasError vrne kot pravilen,
- gumb za prijavo z učinkom ob premiku miške (hover) in zaobljenimi robovi.

Uporabniški vmesnik sem zasnoval tako, da je **intuitiven in preprost** – uporabnik takoj vidi, kaj mora storiti, brez nepotrebnih elementov, ki bi ga motili.



Slika 1: Okno login

(Lastni vir, 2025)



Slika 2: Okno login s prikazom atributa PasswordChar, ki skrije geslo

(Lastni vir, 2025)

3.11 Komponenta Main Window – glavni vmesnik POS

Glavni vmesnik aplikacije je razdeljen na dva glavna dela – **leva stran za izdelke** in **desna stran za košarico**.

MainPosViewModel: to je najkompleksnejši ViewModel v aplikaciji, ki upravlja celotni sistem POS. Vsebuje:

1. Seznam ObservableCollections:

- Products – seznam vseh izdelkov, pridobljenih iz baze,
- CartItems – trenutna košarica z izdelki.

ObservableCollection je posebna vrsta seznama, ki **avtomatsko obvešča UI o spremembah** (dodajanje, odstranjevanje elementov), kar omogoča posodobitev prikaza v realnem času.

2. Izračunljive lastnosti "Computed Properties":

- polje SubtotalExclVAT – osnova brez DDV,
- polje TotalVAT – skupni znesek DDV-ja,
- polje Total – končni znesek za plačilo,
- polje CurrentUser – ime prijavljenega uporabnika.

Te lastnosti se **avtomatsko preračunajo** ob vsaki spremembi košarice, uporabniški vmesnik pa se posodobi preko vezave podatkov.

3. Ukazi:

- AddToCartCommand – doda izdelek v košarico,
- IncreaseQuantityCommand – poveča količino izdelka,
- DecreaseQuantityCommand – zmanjša količino ali odstrani izdelek,
- RemoveFromCartCommand – odstrani celoten izdelek iz košarice,
- ClearCartCommand – počisti celotno košarico,
- ProcessPaymentCommand – procesira plačilo (gotovina ali kartica).

4. **Metoda LoadProductsAsync**: ob inicializaciji komponente ViewModel se asinhrono pokliče _productService.GetAllProductsAsync(), ki pridobi vse aktivne izdelke iz baze. Izdelki se dodajo med izdelke, uporabniški vmesnik pa se avtomatsko posodobi in prikaže seznam.
5. **Logika AddToCart**: ko uporabnik klikne na izdelek ali pritisne gumb "Dodaj v košarico", se izvede naslednja logika:

- preveri, ali izdelek že obstaja v košarici,
  - če obstaja: poveča vrednost Quantity za 1 in preračuna vrednost Total za ta izdelek,
  - če ne obstaja: kreira nov objekt CartItemDto s količino 1 in ga doda v seznam CartItems,

- pokliče metodo RecalculateTotals() za posodobitev skupnih zneskov.

6. **Metoda RecalculateTotals**: to je ključna metoda za finančne izračune:

1. izračuna osnovo brez DDV,
2. izračuna DDV: total – osnovaBrezDDV,
3. sešteje vse osnove → SubtotalExclVAT,
4. sešteje vse DDV → TotalVAT,
5. sešteje vse skupaj → Total.

Vsi izračuni uporabljajo **decimalna števila** za maksimalno natančnost pri finančnih operacijah.

Vrednostne kontrole +/–: ukaz IncreaseQuantityCommand poveča količino za 1, ukaz DecreaseQuantityCommand pa jo zmanjša:

- če je količina večja od 1, samo zmanjša,
- če je količina 1, odstrani celoten artikel iz košarice.

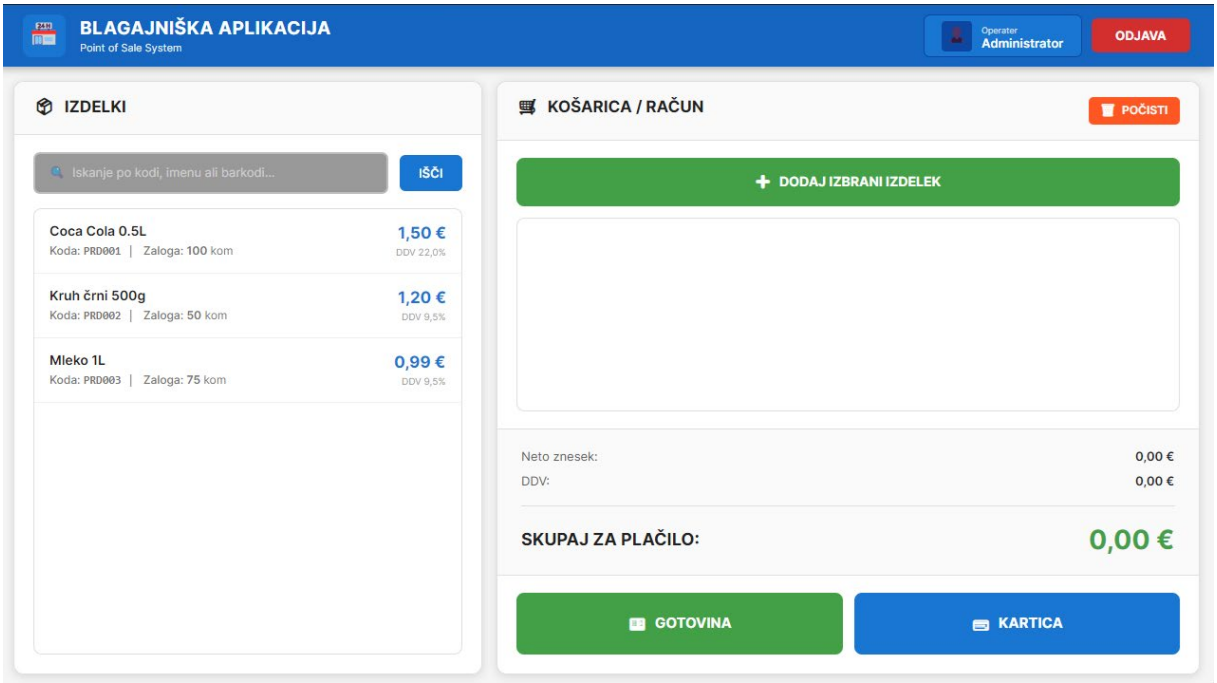
Po vsaki spremembi se pokliče metoda RecalculateTotals() in UI se avtomatsko posodobi.

7. Metoda **ProcessPaymentAsync**: to je najpomembnejša metoda, ki procesira plačilo:

- **validacija**: preveri, ali košarica vsebuje vsaj en artikel. Če ne, vrne napako,
- **kreiranje objekta Receipt**:
  - polje ReceiptNumber: se generira avtomatsko v ReceiptService,
  - polje IssueDateTime: trenutni datum in čas,
  - polje CashierId: ID prijavljenega uporabnika (skladno z GDPR!),
  - polje PaymentMethod: način plačila (gotovina, kartica),
  - polje Total: skupni znesek,
  - polje Status,
- **dodajanje postavk**: za vsak CartItemDto kreira ReceiptItem z vsemi podatki,

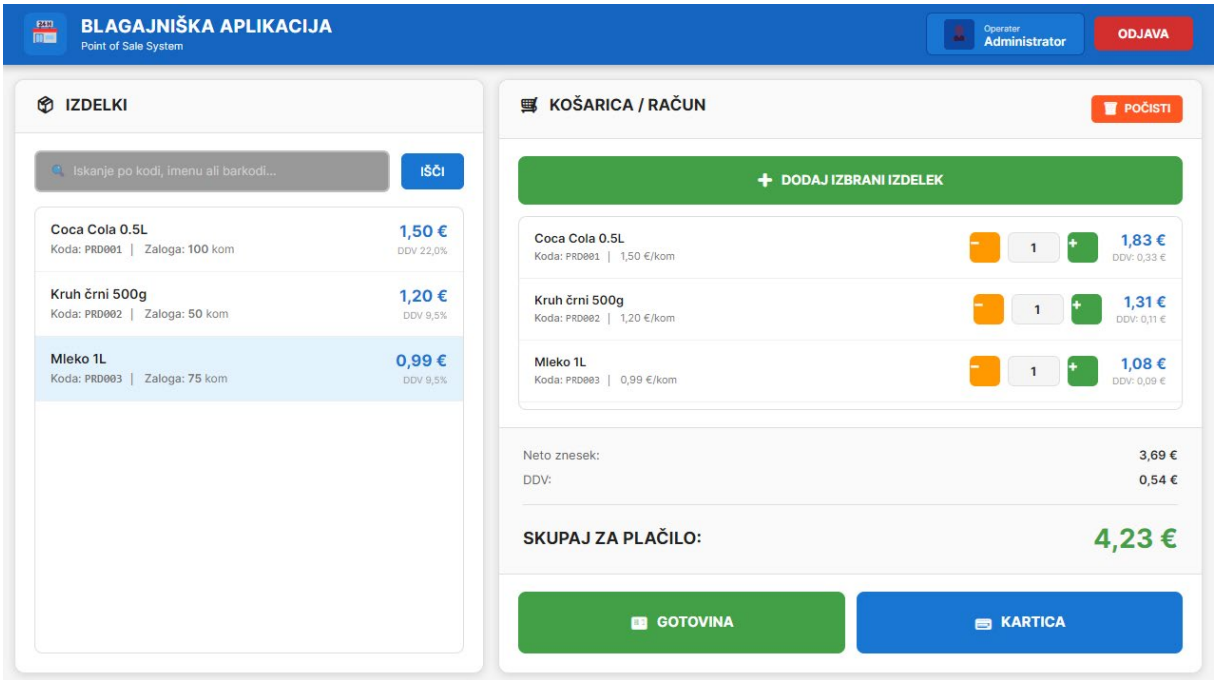
- **shranjevanje v bazo**: pokliče metodo _receiptService.CreateReceiptAsync(receipt),
- **potrditev FURS**: pokliče metodo _fursService.ConfirmReceiptAsync(receipt), ki generira ZOI in EOR,
- **posodobitev v bazi**: posodobi račun s podatki FURS z metodo _receiptService.UpdateReceiptAsync(receipt),
- **prikaz računa**: odpre se okno ReceiptPreviewWindow z vsemi podatki računa,
- **počisti košarico**: pokliče se metoda ClearCart(), kar izprazni košarico in pripravi sistem za novo transakcijo.

Celoten postopek je za obravnavanje napak oziroma izjem, ki v primeru napake nastavi status računa na ReceiptStatus.FursError in prikaže napako uporabniku.



Slika 3: Glavno okno

(Lastni vir, 2025)



Slika 4: Glavno okno s prikazom izbranih izdelkov in skupno ceno

(Lastni vir, 2025)

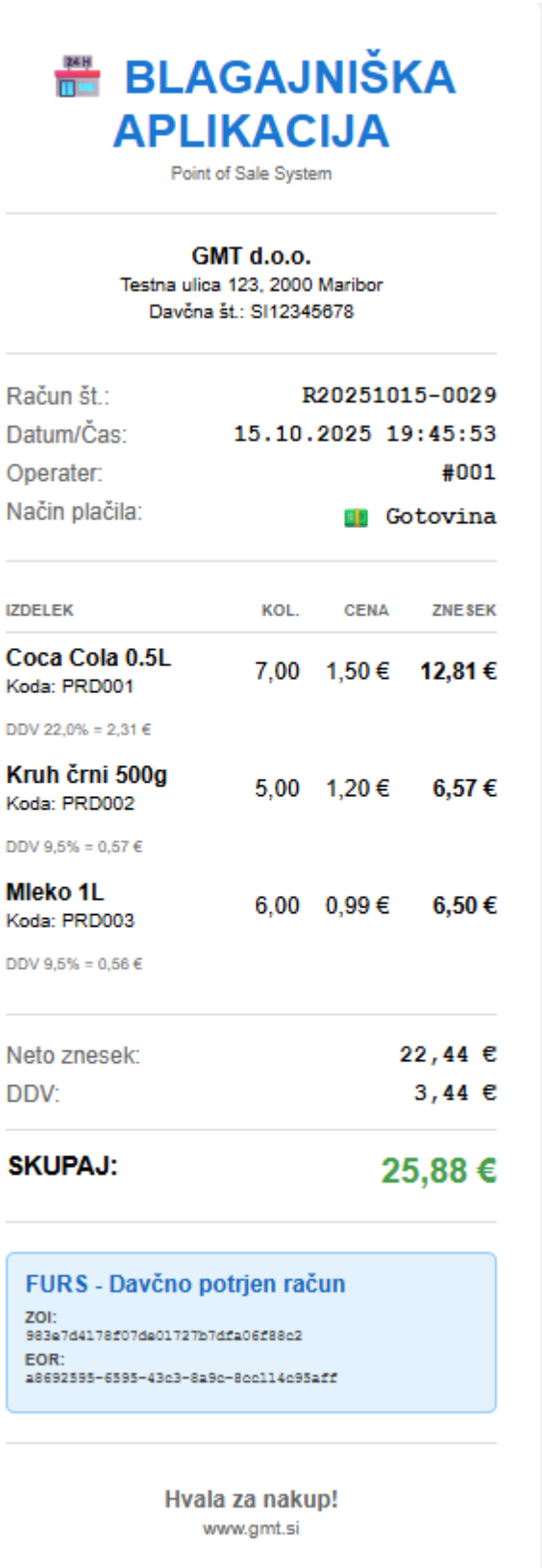
3.12 Integracija FURS – tehnična implementacija

Sistem **FURS** zahteva davčno potrjevanje vseh izdanih računov v skladu z zakonodajo ZDavPR.

Tehnične zahteve:

- šifrirana povezava **TLS 1.2/1.3**,
- **digitalni certifikat** (testni ali produkcijski),
- **format sporočil JSON**,
- **JWS (JSON Web Signature)** za digitalno podpisovanje,
- **dvostopenjski postopek**: najprej ZOI, nato pošiljanje na FURS.

ZOI je niz 32 znakov, sestavljen iz številk (0–9) in črk A–F, ki predstavlja šestnajstiško vrednost, ki služi kot zaščita pred manipulacijo računa.



Slika 5: Primer ZOI in EOR na računu

(Lastni vir, 2025)

3.12 Varnostni mehanizmi

Varnost podatkov je ključnega pomena pri razvoju poslovnih aplikacij, še posebej tistih, ki obdelujejo finančne transakcije in osebne podatke uporabnikov. Čeprav gre v tem primeru za razvojni prototip, sem že v osnovi implementiral ključne varnostne mehanizme, ki bi bili potrebni tudi v produkcijskem okolju.

3.12.1 Varno shranjevanje gesel – šifriranje BCrypt

Eden najpomembnejših varnostnih elementov je **varno shranjevanje uporabniških gesel**. Gesla ne smejo biti nikoli shranjena v berljivi obliki, saj bi v primeru kompromitiranja baze podatkov napadalec pridobil dostop do vseh uporabniških računov.

Za šifriranje gesel sem implementiral **algoritem za šifriranje BCrypt**, ki je trenutno eden izmed najbolj priporočenih algoritmov za ta namen. BCrypt ima več pomembnih lastnosti.

Mehanizem "salt": BCrypt avtomatsko generira naključen **žeton ("salt")** za vsako geslo. Žeton je naključen niz, ki se doda geslu pred šifriranjem. To pomeni, da bosta imeli dve enaki gesli popolnoma različni šifri, kar preprečuje uporabo napadov s pomočjo mavričnih tabel ("rainbow tables"), ki omogočajo hitro iskanje gesel.

Prilagodljiva kompleksnost: BCrypt omogoča nastavitev števila iteracij, kar nadzoruje, kako računsko zahtevno je šifriranje. V implementaciji uporabljam privzeto vrednost 10 iteracij, kar zagotavlja dobro ravnovesje med varnostjo in hitrostjo.

	Id	Username	PasswordHash	FullName	Role	IsActive	CreatedAt	LastLogin	UpdatedAt
1	1	admin	$2a$11$JwLtFn14wgxGzP8zqowJCueK5oYrzbM3mJ.PC5...	Administrator	0	1	0001-01-01 00:00:00.0000000	2025-10-15 17:12:08.0369634	NULL

Slika 6: Primer šifriranega gesla za uporabnika admin

(Lastni vir, 2025)

3.12.2 Skladnost GDPR in minimizacija podatkov

Pri načrtovanju podatkovnega modela sem namenil posebno pozornost uredbi **GDPR (General Data Protection Regulation)**, ki vsebuje naslednje zahteve.

Minimizacija osebnih podatkov: na računih shranjujem **samo ID blagajnika** (CashierId), ne pa tudi polnega imena ali drugih osebnih podatkov. To pomeni:

- račun vsebuje samo številčni ID (npr. 5),
- polno ime je dostopno samo preko tabele Users,
- na natisnjenem računu piše "Operater: ID 5",
- v primeru zahteve GDPR po izbrisu lahko deaktiviram uporabnika (IsActive = false), računi pa ostanejo veljavni.

Namen omejitev: podatke zbiram in shranjujem **samo za zakonsko potrebne namene**:

- podatki o računih: potrebni za davčno zakonodajo (7 let hrambe),
- podatki o uporabnikih: potrebni za avtentikacijo in sledljivost,
- podatki o kupcih: **ne shranjujem**, razen če je potrebna faktura DDV za pravne osebe.

Pravica do izbrisa: implementiral sem pristop **"soft delete"**:

- uporabniki se ne izbrišejo fizično iz baze,
- nastavi se IsActive = false,
- deaktivirani uporabniki se ne morejo prijaviti,
- zgodovinski računi ostanejo ohranjeni (zakonska zahteva).

3.12.3 Zaščitena komunikacija – HTTPS/TLS

Za komunikacijo z zunanjimi sistemi (FURS API) je obvezna uporaba protokola **TLS 1.2 ali 1.3**, kar zagotavlja:

- **šifriranje prenosa**: vsi podatki, ki potujejo med aplikacijo in strežnikom FURS, so šifrirani. To preprečuje napade, t. i. "man-in-the-middle", kjer bi napadalec lahko prestrezal in prebral občutljive podatke;
- **avtentikacija strežnika**: certifikat TLS zagotavlja, da komuniciram z legitimnim strežnikom FURS, ne pa z napadalčevim lažnim strežnikom;
- **integriteta podatkov**: TLS zagotavlja, da podatki niso bili spremenjeni med prenosom.

3.12.4 Avtorizacija in nadzor dostopa

Implementiral sem **role-based access control (RBAC)**.

Vloge uporabnikov:

- **Admin**: polni dostop do sistema, upravljanje uporabnikov in izdelkov;
- **Cashier**: dostop samo do vmesnika POS, ne more upravljati nastavitev.

Časovna omejitev seje: čeprav trenutno ni implementirano, bi v produkcijskem okolju dodal:

- avtomatsko odjavo po 30 minutah neaktivnosti,
- ponovno avtentikacijo pri občutljivih operacijah,
- beleženje vseh prijav in pomembnih akcij.

3.12.5 Preprečevanje injekcije SQL

Uporaba **Entity Framework Core** avtomatsko zagotavlja zaščito pred napadi SQL:

- **parametrizirane poizvedbe**: EF Core uporablja parametre namesto direktnega vstavljanja uporabniških vnosov v stavke SQL:

```
-- Napačno (ranljivo):
SELECT * FROM Users WHERE Username = 'admin'

-- Pravilno (EF Core):
SELECT * FROM Users WHERE Username = @p0
Parameter: @p0 = 'admin'
```

- **LINQ queries**: namesto pisanja SQL-ja uporabljam LINQ:

```
var user = await _context.Users
    .FirstOrDefaultAsync(u => u.Username == username)
```

EF Core to avtomatsko prevede v varno parametrizirano poizvedbo SQL.

Validacija vhodnih podatkov: kljub zaščiti EF Core sem implementiral dodatno validacijo:

- preverjanje praznih polj,
- omejevanje dolžine vnosov (MaxLength),
- preverjanje formatov (elektronski naslov, telefonska številka),
- sanitizacija vnosov (odstranjevanje nevarnih znakov).

3.12.6 Obravnava napak in logiranje

Bloki "Try-Catch": vse kritične operacije sem ovil v bloke try-catch:

```
try
{
   // Operacija z bazo, klic FURS itd.
}
catch (SqlException ex)
{
   // Specifično za napake database
   _logger.LogError("Database error: " + ex.Message);
   ShowError("Napaka pri dostopu do baze podatkov");
}
```

```
catch (Exception ex)
{
    // Splošne napake
    _logger.LogError("Unexpected error: " + ex.Message);
    ShowError("Prišlo je do nepričakovane napake");
}
```

Za produkcijo bi dodal:

- **beleženje dogodkov (File logging)** – uporaba knjižnic Serilog ali NLog za zapisovanje dogodkov aplikacije v datoteke, kar omogoča sledenje napakam in analizo delovanja sistema;
- **nivoji beleženja (Log levels)** – razlikovanje med petimi nivoji: Debug (razhroščevanje), Info (informativna sporočila), Warning (opozorila), Error (napake) in Critical (kritične napake), kar omogoča filtriranje in prioritizacijo sporočil;
- **rotacija dnevniških datotek (Log rotation)** – avtomatsko ustvarjanje novih dnevniških datotek (dnevno ali tedensko) za preprečevanje prevelikih datotek in lažje arhiviranje;
- **centralizirano beleženje** – v primeru več naprav poteka zbiranje vseh dnevnikov na enem strežniku, kar omogoča celovit pregled nad delovanjem celotnega sistema in hitrejše odkrivanje težav.

3.12.7 Potencialne prihodnje izboljšave varnosti

Dvofaktorska avtentikacija (2FA):

- **enkratna gesla (OTP – One-Time Password)** – sistem pošlje uporabniku enkratno veljavno kodo preko elektronske pošte ali sporočila SMS, ki jo mora vnesti poleg običajnega gesla za dostop do aplikacije;
- **avtentikacijska aplikacija (Authenticator app)** – uporaba mobilnih aplikacij, kot sta Google Authenticator ali Microsoft Authenticator, ki generirajo časovno omejene kode (običajno 30 sekund), neodvisno od internetne povezave;

- **strojni ključ (Hardware token)** – fizična naprava (USB ključek ali podobno), ki generira enkratne kode ali se neposredno poveže z računalnikom za potrditev identitete, kar zagotavlja najvišjo raven varnosti.

Šifriranje občutljivih podatkov v bazi:

- šifriranje AES-256 za občutljive podatke,
- ključi shranjeni ločeno od baze,
- transparentno šifriranje podatkov na strežniku SQL.

Omejitev števila zahtev:

- omejitev števila poskusov prijave (npr. 5 neuspešnih = 15 min lockout),
- preverjanje napadov brute-force,
- CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) po več neuspešnih poskusih.

Dnevnik sprememb:

- beleženje vseh pomembnih akcij (kdo, kaj, kdaj),
- nespremenljiv log,
- redne revizije.

Pripenjanje certifikatov:

- pri komunikaciji FURS preveriti točno določen certifikat,
- preprečuje napade MITM z lažnimi certifikati.

4 ANKETIRANJE

Za validacijo postavljenih hipotez in evalvacijo razvite aplikacije POS je bila izvedena **uporabniška anketa** med potencialnimi uporabniki sistema. Cilj raziskave je bil preveriti uporabniško izkušnjo, zadovoljstvo z vmesnikom, zaupanje v varnostne mehanizme in pripravljenost za uporabo razvite rešitve.

4.1 Metodologija raziskave

Vzorec: v anketi je sodelovalo **48 anketirancev**, od katerih je bilo **47 odgovorov veljavnih** za analizo.

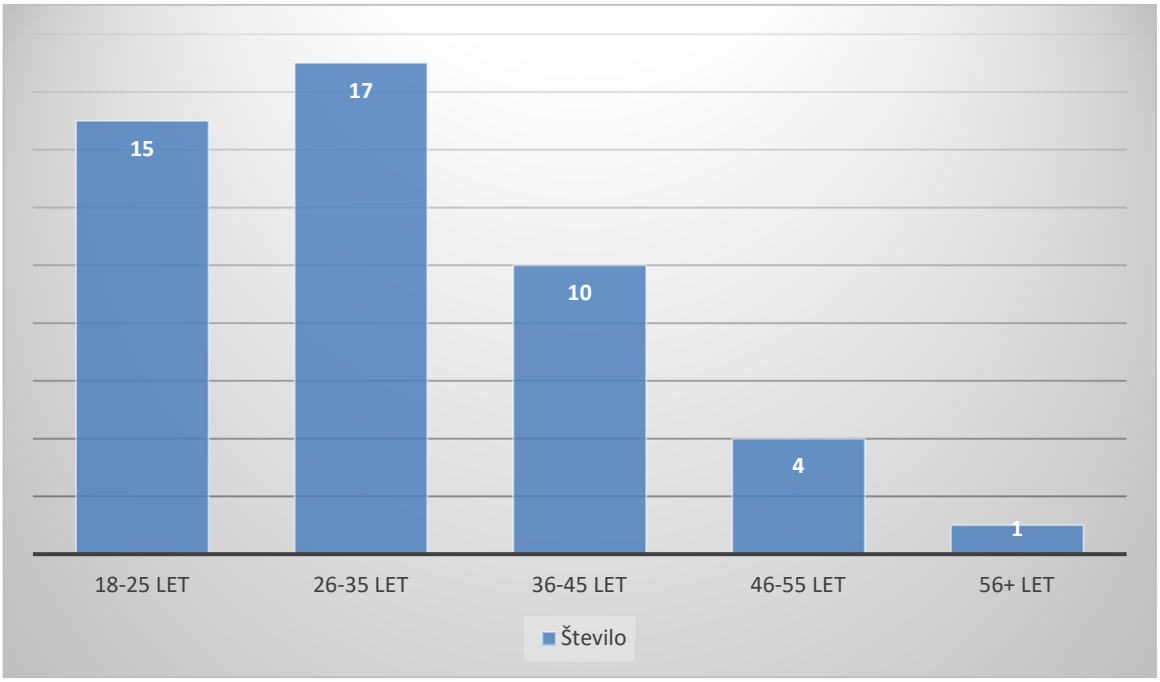
Metoda zbiranja podatkov: spletna anketa (1ka.si).

Obdobje izvedbe: oktober 2025.

Ciljna skupina: potencialni uporabniki sistemov POS – lastniki manjših podjetij, samostojni podjetniki, zaposleni v trgovinah in gostinstvu, ter tehnično usposobljeni uporabniki.

4.2 Demografska struktura vzorca

4.2.1 Starostna struktura anketirancev

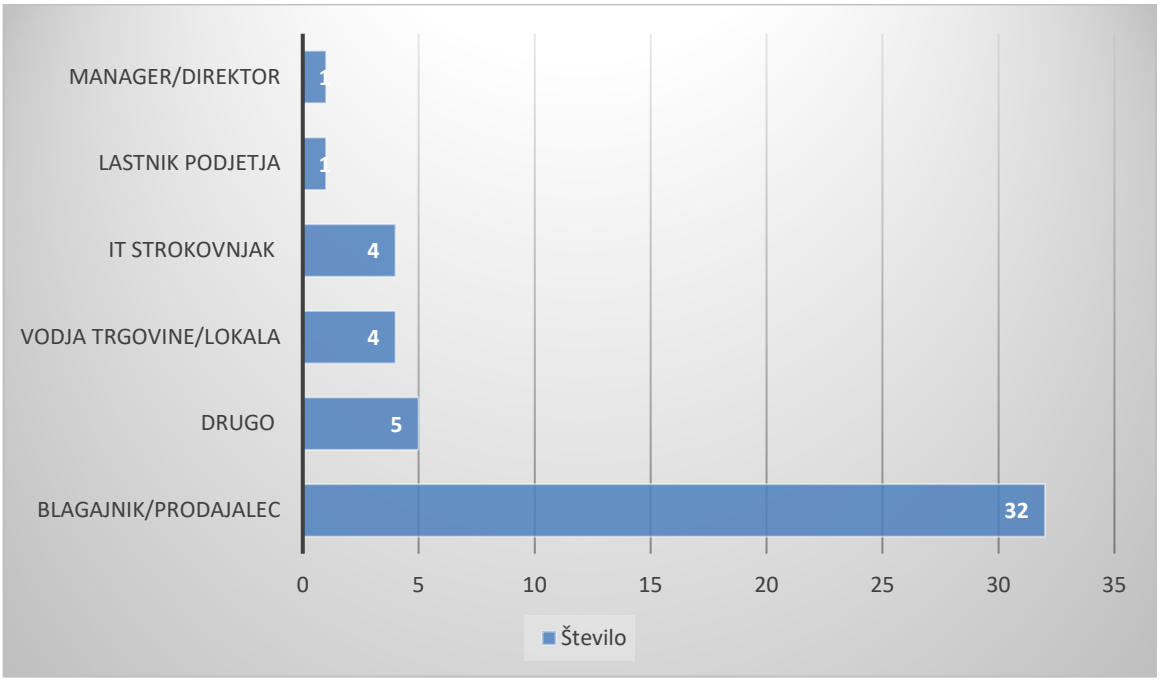


Graf 1: Starostna struktura anketirancev

(Lastni vir, 2025)

Največ anketirancev je starih od 18 do 35 let (68 %), torej gre za mlajšo generacijo, ki je odraščala z računalniki in pametnimi telefoni. Največ jih je v skupini 26–35 let (36 %), kar pomeni, da imajo že nekaj let delovnih izkušenj in so idealni za testiranje take aplikacije. Mlajših od 25 let je bilo 32 %, kar kaže, da je aplikacijo testirala tudi generacija, ki je že od malega navajena hitrega učenja novih programov. Starejših uporabnikov je bilo precej manj, v skupini od 36 do 45 let jih je bilo 21 %, nad 46 let pa je imelo 11 % vseh anketirancev.

4.2.2 Delovno mesto anketirancev

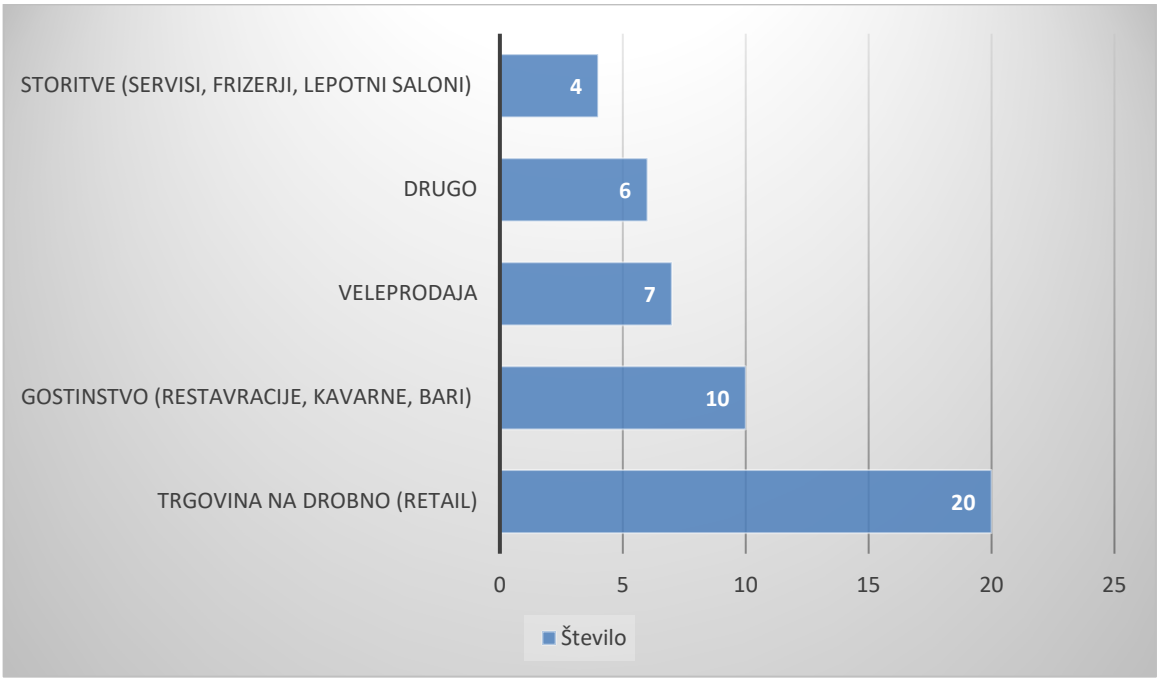


Graf 2: Delovno mesto anketirancev

(Lastni vir, 2025)

Večina anketirancev (68 %) dela kot blagajnik ali prodajalec, kar je ključno za raziskavo, saj so to ljudje, ki bi aplikacijo dejansko uporabljali vsak dan. Precej jih je tudi v kategoriji Drugo (11 %), kar verjetno pomeni, da so študenti ali ljudje, ki so aplikacijo testirali iz radovednosti. Vodje trgovin in strokovnjaki IT so bili zastopani z enakim številom (po 4 osebe), medtem ko sta bila lastnik podjetja in manager zastopana z le eno osebo. Ta sestava vzorca je idealna za testiranje aplikacije, ker daje povratne informacije od tistih, ki sistem res potrebujejo in uporabljajo.

4.2.3 Sektor dela anketirancev



Graf 3: Sektor dela anketirancev

(Lastni vir, 2025)

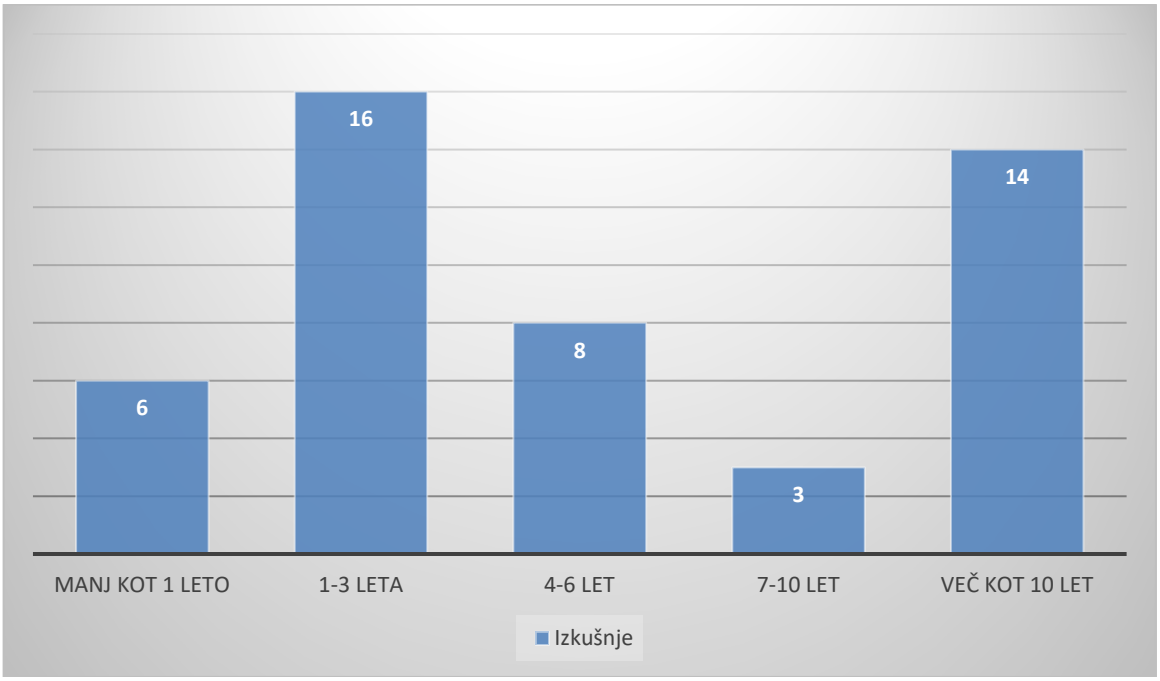
Pod Drugo (6 oseb) spadajo:

- računovodstvo: 1,
- dom za ostarele: 1,
- industrija: 1,
- zavarovalnica (klicni center): 1,
- razvijalec: 1,
- en manjkajoč odgovor.

Največ ljudi, ki so izpolnili anketo, dela v trgovinah (43 %). To je logično, ker trgovine najbolj potrebujejo blagajniške sisteme. Kar nekaj jih dela tudi v gostinstvu – restavracijah, kavarnah in barih (21 %). Precej jih je tudi iz veleprodaje (15 %) in različnih storitvenih dejavnosti, kot

so frizerji in kozmetični saloni (9 %). Drugi delajo v različnih panogah, od računovodstva do industrije in zavarovalnic.

4.2.4 Leta izkušenj z uporabo sistemov POS

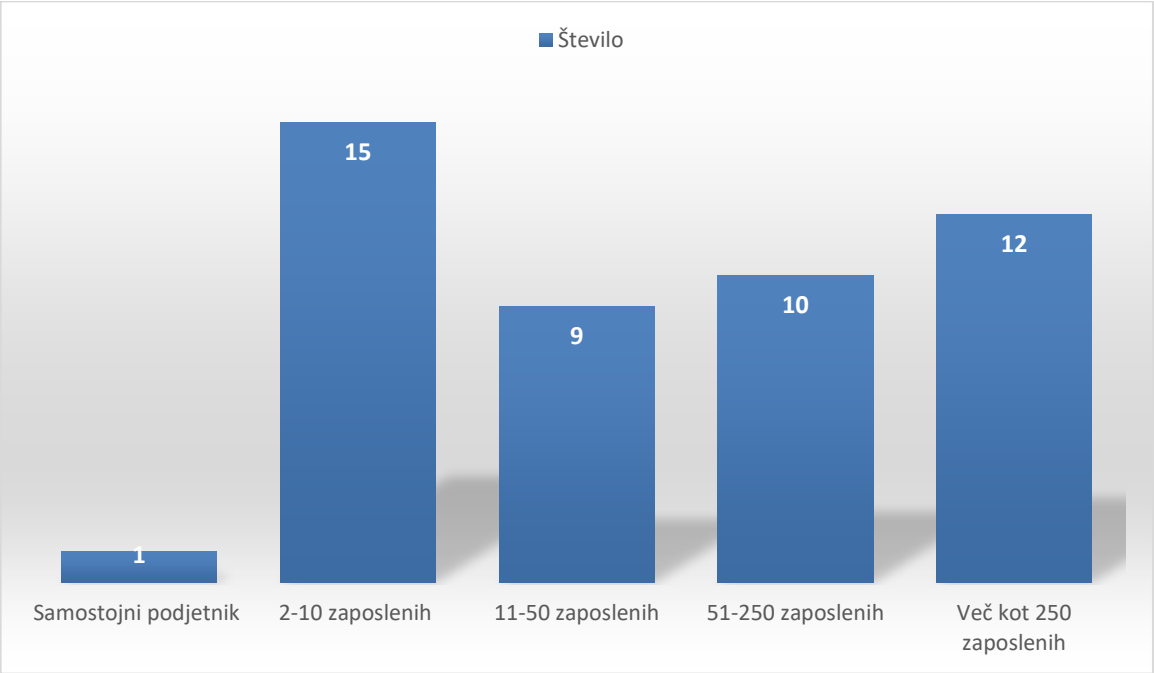


Graf 4: Leta izkušenj z uporabo sistemov POS

(Lastni vir, 2025)

Anketiranci imajo zelo različne izkušnje z uporabo sistemov POS. Največ jih ima **1–3 leta izkušenj** (34 %), kar kaže, da so relativno novi uporabniki. Kar 30 % pa ima **več kot 10 let izkušenj**, torej gre za zelo izkušene uporabnike, ki poznajo različne sisteme. Skupina s 4–6 let izkušenj predstavlja 17 %, popolni začetniki z manj kot letom dni pa predstavljajo 13 %. Najmanj je tistih s 7–10 let izkušenj (6 %).

4.2.5 Velikost podjetja anketirancev



Graf 5: Velikost podjetja anketirancev

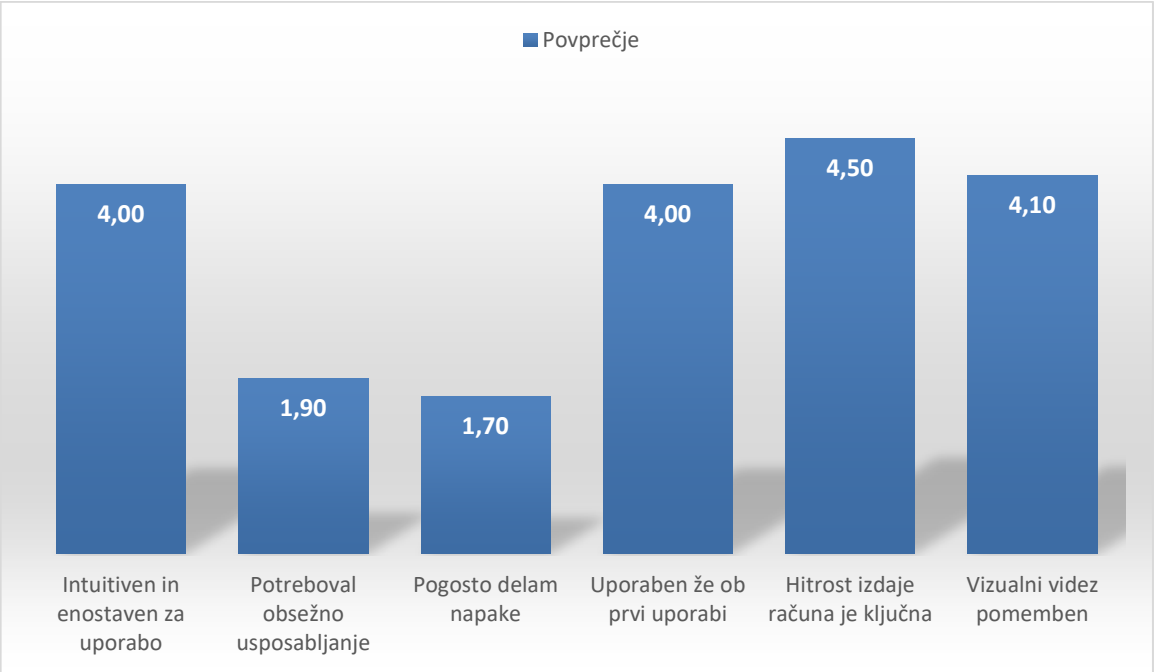
(Lastni vir, 2025)

Anketiranci prihajajo iz podjetij različnih velikosti, kar daje dobro reprezentativno sliko trga. Največ jih dela v **majhnih podjetjih z 2–10 zaposlenimi** (32 %), kar je tipično za slovensko gospodarstvo. Sledijo **velika podjetja z več kot 250 zaposlenimi** (26 %), potem **srednje velika podjetja z 51–250 zaposlenimi** (21 %) in **manjša podjetja z 11–50 zaposlenimi** (19 %). Samostojni podjetnik je bil zastopan le z eno osebo (2 %).

To je zelo pomembno za validacijo **hipoteze H3** (lokalne rešitve so stroškovno učinkovitejše za mala podjetja), saj imamo v vzorcu dovolj ljudi iz manjših podjetij, ki so glavna ciljna skupina za tako aplikacijo.

4.3 Uporabniška izkušnja

4.3.1 Ocena uporabniške izkušnje

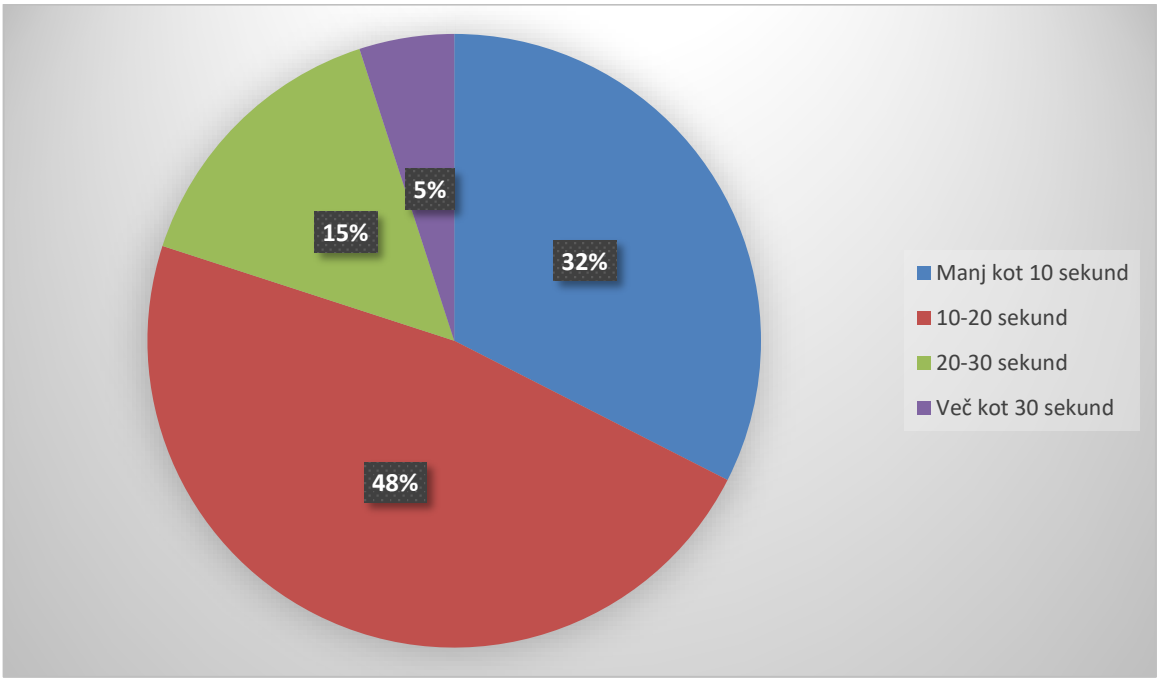


Graf 6: Ocena uporabniške izkušnje

(Lastni vir, 2025)

Anketiranci so na lestvici od 1 do 5 ocenili svoja **pričakovanja in izkušnje s sistemi POS na splošno**. Rezultati kažejo, da je **hitrost izdaje računa** najpomembnejša zahteva (4,5), kar potrjuje, da morajo biti sistemi POS izjemno hitri. Visoko so ocenili tudi pomembnost **intuitivnosti** (4,0) in **videza** (4,1). Pozitivno je, da uporabniki menijo, da **ne bi smeli potrebovati obsežnega usposabljanja** (1,9 – nizka ocena pomeni, da se ne strinjajo s potrebo po usposabljanju) in da **redko delajo napake** zaradi zapletenih vmesnikov (1,7). Ti rezultati potrjujejo **hipotezo H1** – uporabniki res pričakujejo hitro in intuitivno rešitev, ki deluje že ob prvi uporabi, brez dolgega učenja.

4.3.2 Potrebni čas za izdajo računa



Graf 7: Potrebni čas za izdajo računa

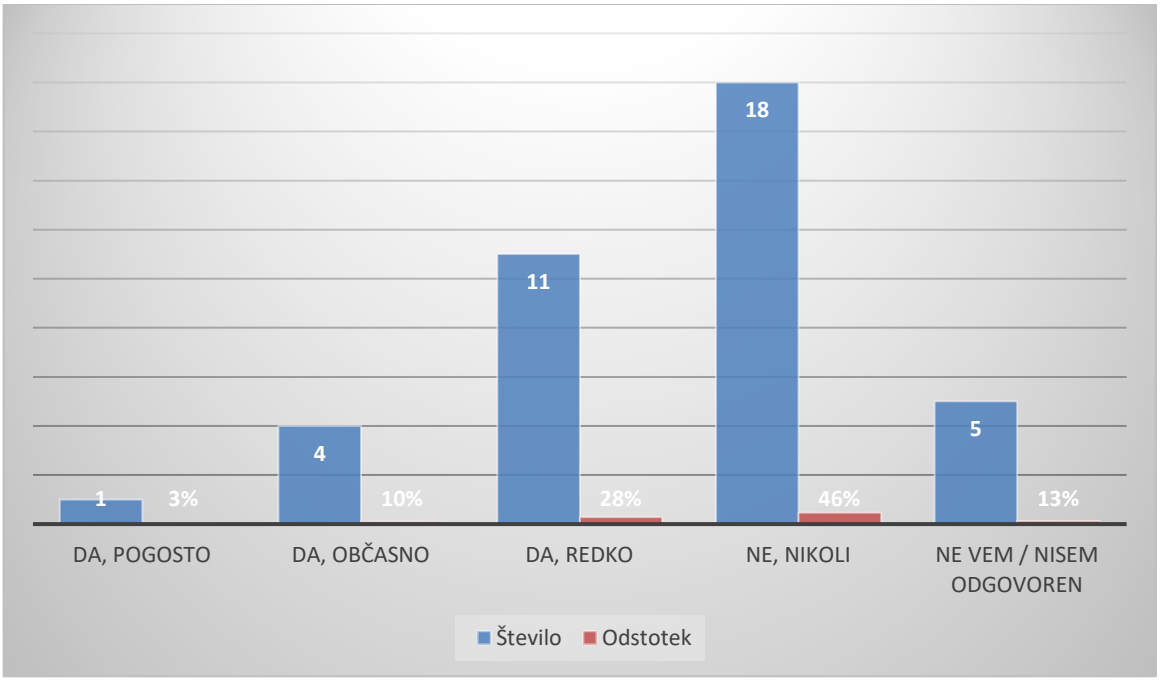
(Lastni vir, 2025)

Večina uporabnikov (48 %) potrebuje **10–20 sekund** za izdajo enega računa, kar kaže na solidno hitrost dela. Kar 33 % uporabnikov je še hitrejših in izdajo račun v **manj kot 10 sekundah**, kar je odlično. Skupaj kar **81 % uporabnikov** izda račun v **manj kot 30 sekundah**, kar je zelo hitra obdelava. Samo 5 % potrebuje več kot 30 sekund, kar je sprejemljivo.

Ti rezultati potrjujejo, da je **hitrost ključna zahteva** sistemov POS – večina uporabnikov že dela zelo hitro in pričakuje, da jim sistem to tudi omogoča. To je v skladu z visoko oceno iz prejšnjega vprašanja, kjer je hitrost izdaje računa dobila oceno 4,5/5.

4.4 Integracija FURS

4.4.1 Pogostost težav z integracijo FURS



Graf 8: Pogostost težav z integracijo FURS

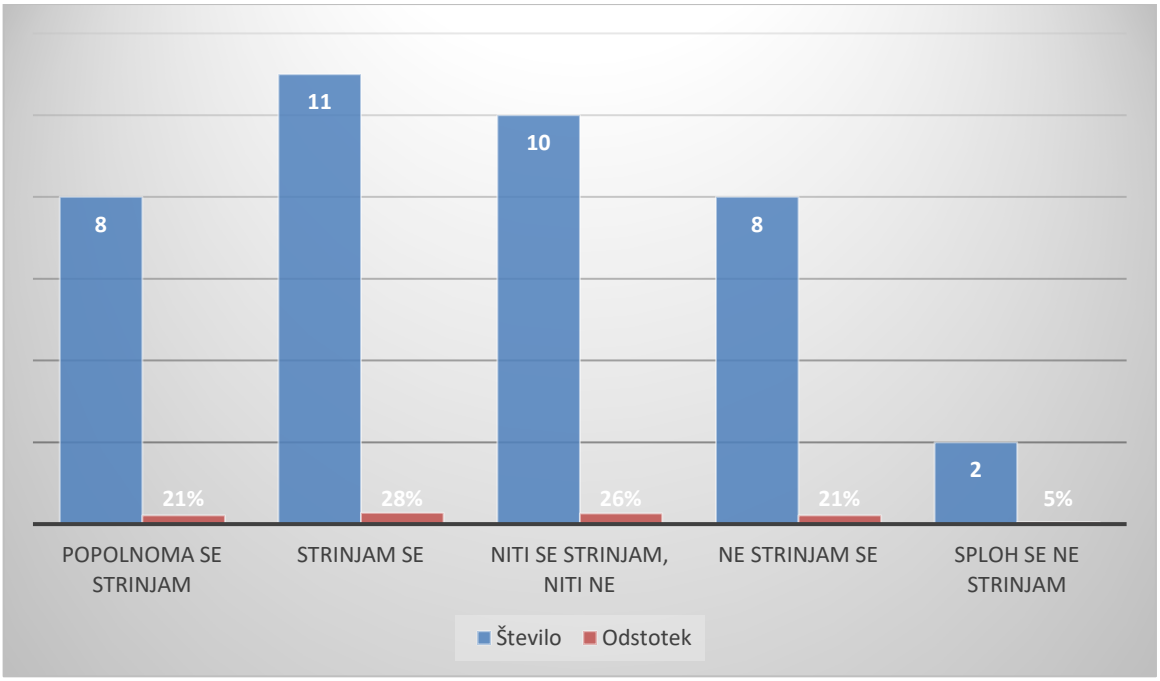
(Lastni vir, 2025)

Skoraj **polovica uporabnikov (46 %) nikoli ni imela težav** s potrjevanjem računov FURS, kar kaže na relativno zanesljivo delovanje sistema. Če tej skupini prištejemo še tiste, ki težave doživljajo le redko (28 %), dobimo kar **74 % uporabnikov brez resnih težav**. Občasne težave ima 10 % uporabnikov, pogoste pa le 3 %. Zanimivo je, da 13 % uporabnikov ne ve, ali niso odgovorni za to funkcionalnost, kar nakazuje, da se z integracijo FURS ukvarjajo predvsem strokovnjaki IT ali vodje.

Kljub relativno pozitivnim rezultatom pa podatek, da je **več kot četrtina uporabnikov** (28 % + 10 % + 3 % = 41 %) doživela vsaj nekatere težave, potrjuje **hipotezo H2** – zanesljiva

integracija FURS je res ključna, saj lahko že občasne težave povzročijo motnje v poslovanju in nezadovoljstvo uporabnikov.

4.4.2 Odvisnost poslovanja od sistema FURS



Graf 9: Odvisnost poslovanja od sistema FURS

(Lastni vir, 2025)

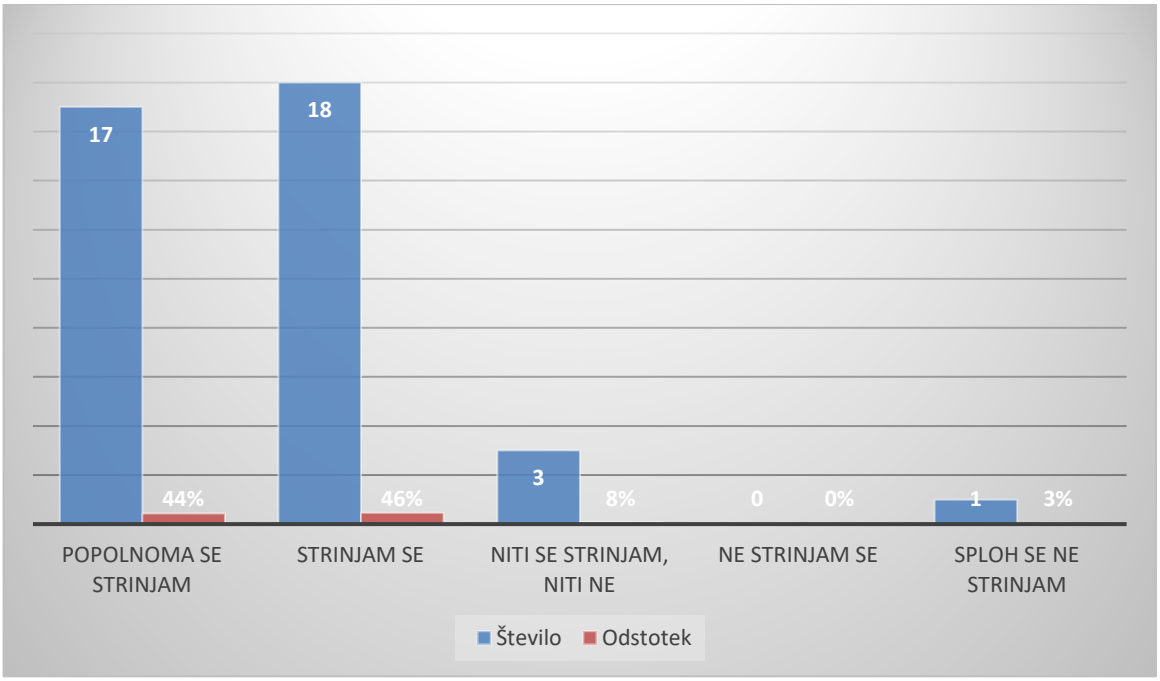
Rezultati kažejo **mešana mnenja** o sposobnosti dela brez sistema FURS. Skoraj **polovica uporabnikov (49 %)** meni, da lahko **nemoteno nadaljujejo z delom** tudi ob izpadu sistema FURS (21 % + 28 %), kar verjetno pomeni, da njihovi sistemi omogočajo način brez internetne povezave ali odloženo pošiljanje.

Po drugi strani pa kar **26 % uporabnikov** ni prepričanih, dodatnih **21 % se ne strinja** in **5 % sploh ne soglaša**, da bi lahko delali nemoteno. Skupaj je torej **52 % uporabnikov**, ki so vsaj delno **odvisni od delovanja sistema FURS**.

Ti rezultati dodatno potrjujejo **hipotezo H2** – zanesljiva integracija FURS je kritična, saj več kot polovica uporabnikov ne more (ali ni prepričana, če lahko) normalno poslovati ob izpadu

sistema. To pomeni, da morajo rešitve POS vključevati **mehanizme za delo v načinu brez internetne povezave** ali vsaj začasno shranjevanje računov za poznejše pošiljanje.

4.4.3 Vrednost avtomatske integracije FURS



Graf 10: Vrednost avtomatske integracije FURS

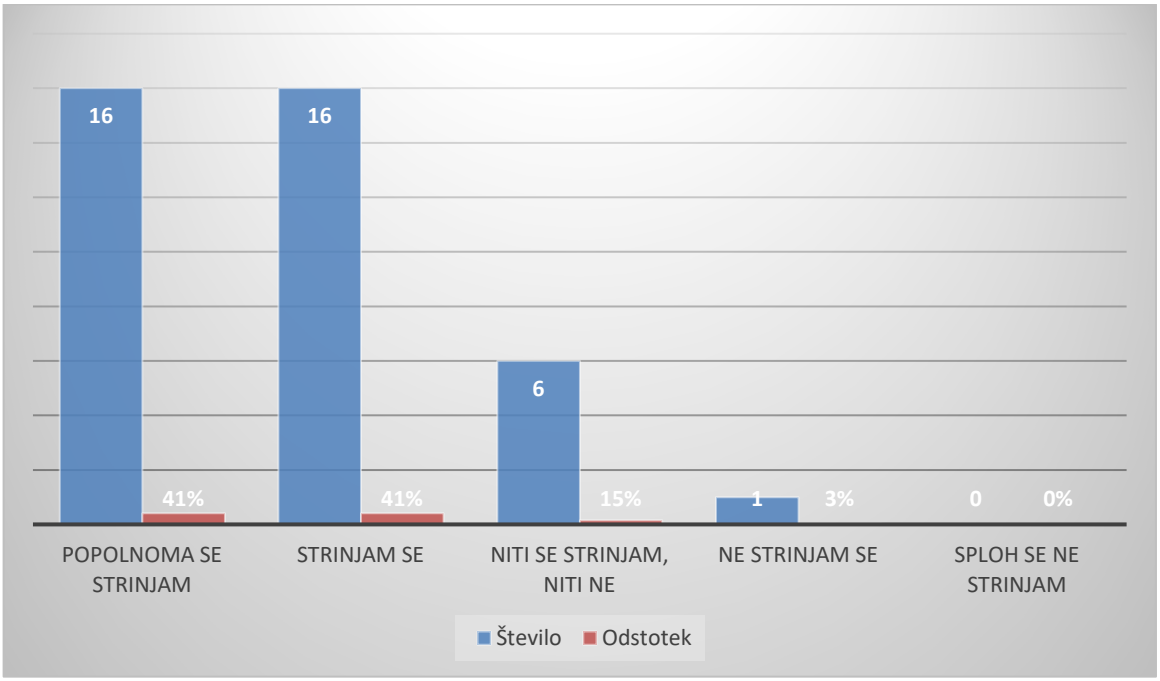
(Lastni vir, 2025)

Kar **90 % uporabnikov** (44 % + 46 %) se **strinja ali popolnoma strinja**, da jim avtomatsko davčno potrjevanje prihrani čas in zmanjša možnost napak. To je **zelo visoka stopnja zadovoljstva** in jasno kaže, da uporabniki cenijo avtomatizacijo tega procesa.

Samo **3 % se sploh ne strinjajo**, medtem ko nihče ni izbral možnosti "Ne strinjam se". To pomeni, da je **integracija FURS praktično univerzalno zaželena funkcionalnost** med uporabniki sistemov POS.

Ti rezultati **močno potrjujejo hipotezo H2** – zanesljiva integracija z davčnim potrjevanjem FURS ni samo zakonska zahteva, ampak tudi **resnična dodana vrednost** za uporabnike, ki jim prihrani čas, zmanjša administrativno breme in prepreči napake pri ročnem vnosu.

4.4.4 Kritičnost zanesljive integracije FURS



Graf 11: Zanesljivost integracije FURS

(Lastni vir, 2025)

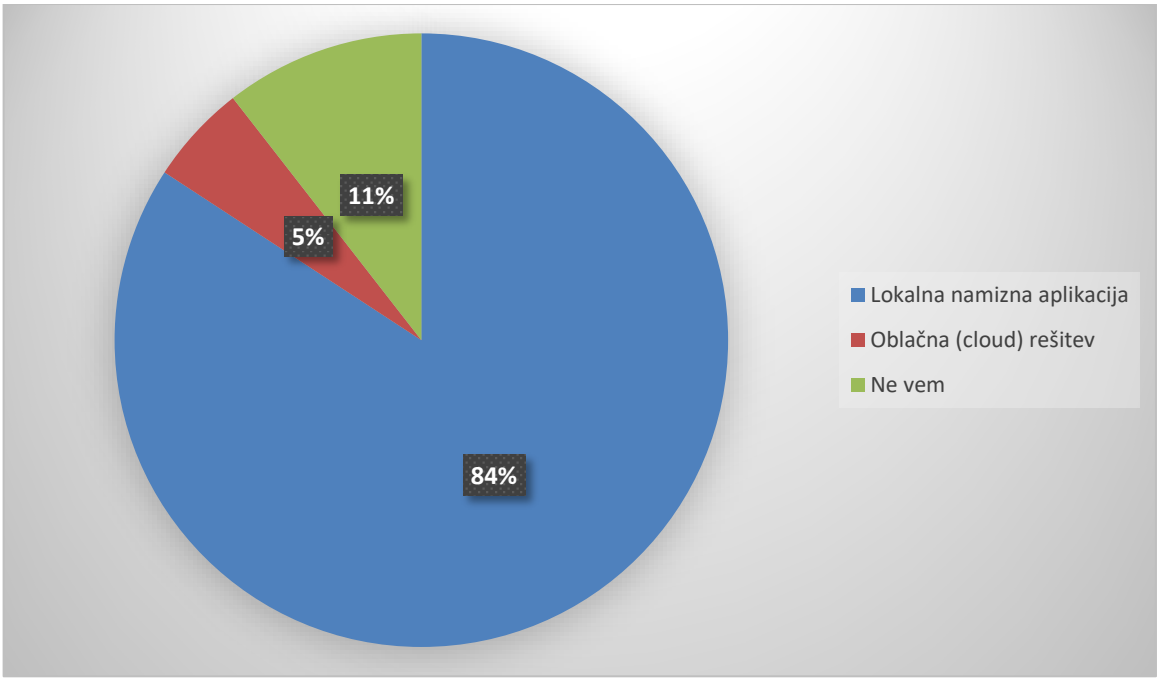
To vprašanje prinaša **izjemno jasne rezultate**, ki **neposredno potrjujejo hipotezo H2**. Kar **82 % uporabnikov** (41 % + 41 %) se **strinja ali popolnoma strinja**, da je zanesljiva integracija s sistemom FURS **ključna za nemoteno delovanje blagajne**. To je zelo visoka stopnja soglasja.

Samo **15 % uporabnikov** je nevtralnih, medtem ko se le **3 % ne strinjajo**. Nihče ni izbral možnosti "Sploh se ne strinjam", kar pomeni, da **praktično vsi uporabniki** priznavajo pomembnost zanesljive integracije FURS.

Ti rezultati v kombinaciji s prejšnjimi vprašanji (kjer 90 % uporabnikov ceni avtomatizacijo) **nedvomno potrjujejo hipotezo H2 – zanesljiva integracija z davčnim potrjevanjem FURS je resnično ključna za nemoteno delovanje blagajniškega sistema** in predstavlja kritično funkcionalnost, ki jo uporabniki pričakujejo od sodobnih rešitev POS.

4.5 Stroškovna učinkovitost

4.5.1 Trenutna uporaba lokalnih oziroma oblačnih rešitev



Graf 12: Trenutna uporaba sistema POS

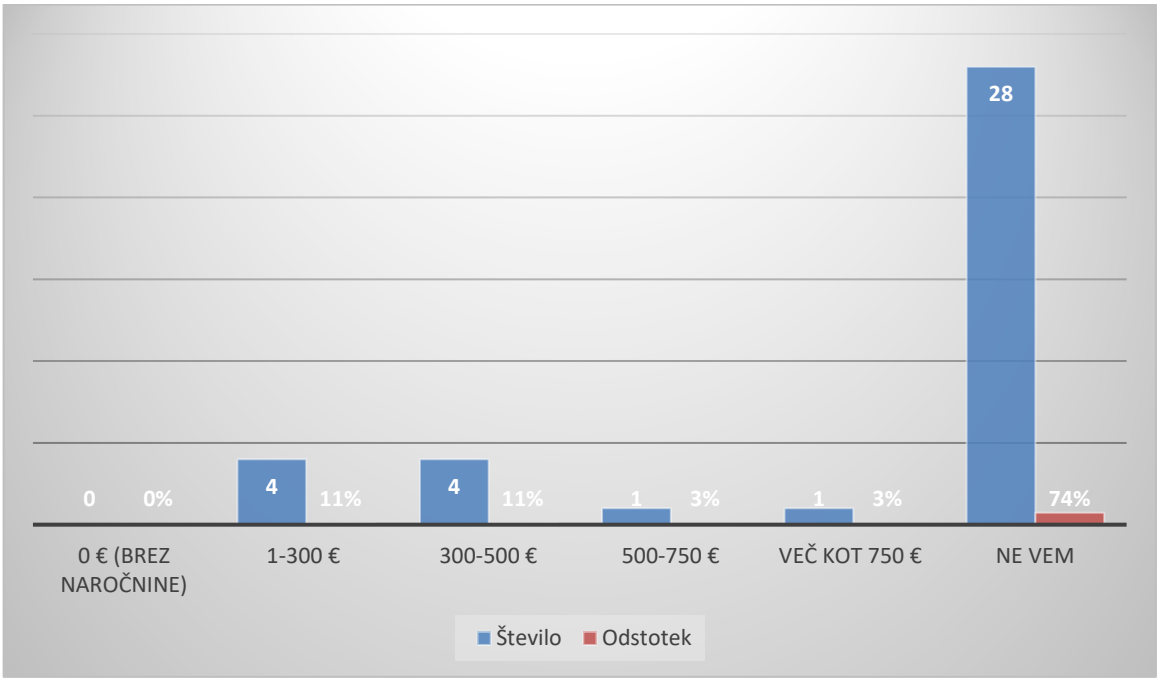
(Lastni vir, 2025)

Rezultati jasno kažejo **prevlado lokalnih namiznih aplikacij** – kar **84 % uporabnikov** trenutno uporablja lokalno nameščene sisteme POS, medtem ko jih le **5 % uporablja oblačne rešitve**. To je zelo pomemben podatek, ki kaže, da so **lokalne rešitve še vedno dominantne** na trgu, še posebej v manjših podjetjih.

Zanimivo je, da **11 % uporabnikov ne ve**, kakšen tip sistema uporabljajo, kar nakazuje, da se odločitve IT pogosto sprejemajo na višjih nivojih in končni uporabniki (blagajniki) niso vedno seznanjeni s tehnološkimi podrobnostmi.

Ta visoka stopnja uporabe lokalnih rešitev **podpira hipotezo H3**, saj kaže, da podjetja v praksi že zdaj **preferirajo lokalne sisteme** pred oblačnimi. V kombinaciji z naslednjimi vprašanji o stroških bomo lahko potrdili, ali je glavni razlog za to res **stroškovna učinkovitost**.

4.5.2 Trenutni mesečni stroški za sisteme POS



Graf 13: Mesečni stroški za uporabo sistema POS

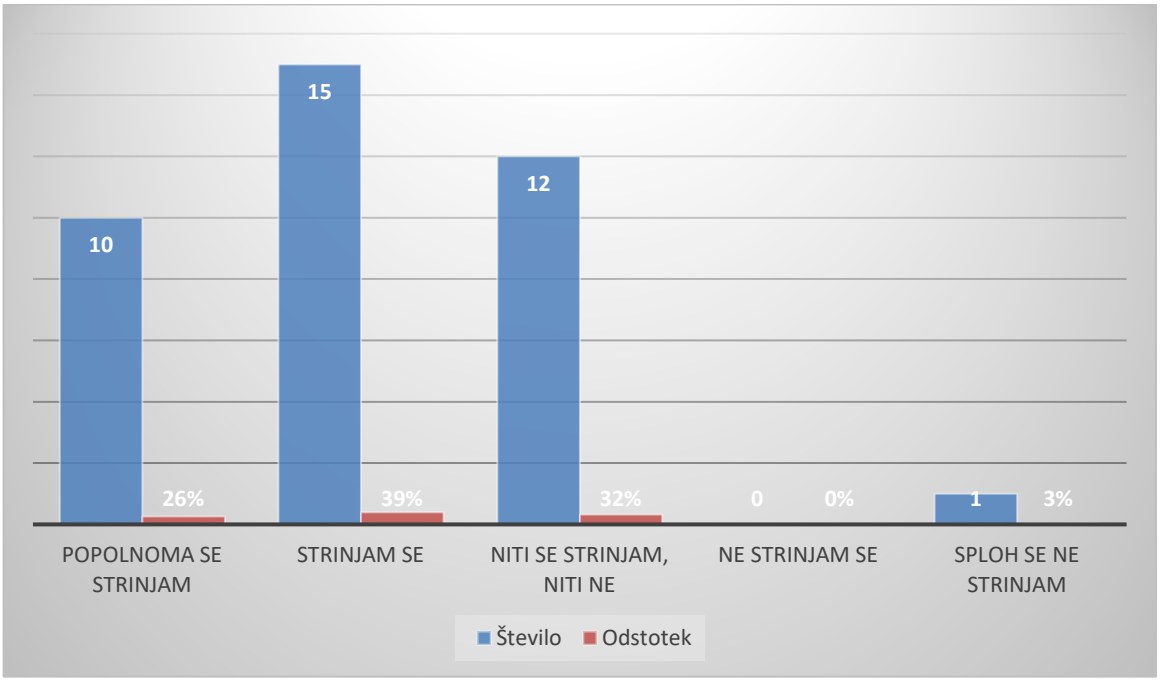
(Lastni vir, 2025)

Zelo zanimiv in **presenetljiv rezultat** – kar **74 % uporabnikov ne ve**, koliko njihovo podjetje plačuje za sistem POS. To kaže, da večina **končnih uporabnikov (blagajnikov) ni vključenih v finančne odločitve** glede nakupa in vzdrževanja sistemov.

Od tistih, ki poznajo stroške (26 %), jih **večina plačuje 1–500 € mesečno** (11 % + 11 % = 22 %). Le **3 % plačujejo več kot 500 € mesečno**, kar je razmeroma visok strošek.

Pomemben zaključek: ker kar 84 % uporablja **lokalne sisteme** (iz prejšnjega vprašanja) in večina ne plačuje visokih mesečnih naročnin, to **podpira hipotezo H3**, da so **lokalne rešitve stroškovno ugodnejše** od oblačnih sistemov z mesečnimi naročninami. Mnogi uporabniki verjetno plačajo enkratno licenco ali imajo minimalne mesečne stroške vzdrževanja.

4.5.3 Percepcija stroškovne učinkovitosti lokalnih rešitev za mala podjetja



Graf 14: Stroškovna učinkovitost lokalnih rešitev

(Lastni vir, 2025)

To vprašanje prinaša **ključne rezultate za validacijo hipoteze H3**. Kar **65 % uporabnikov** (26 % + 39 %) se **strinja ali popolnoma strinja**, da so **lokalne namizne aplikacije brez mesečne naročnine stroškovno učinkovitejše za majhna podjetja**. To je zelo visoka stopnja soglasja.

Pomembno je, da **nihče ni izbral možnosti "Ne strinjam se"** in samo **3 % se sploh ne strinjajo**. To pomeni, da je prepričanje o stroškovni prednosti lokalnih rešitev **praktično univerzalno**.

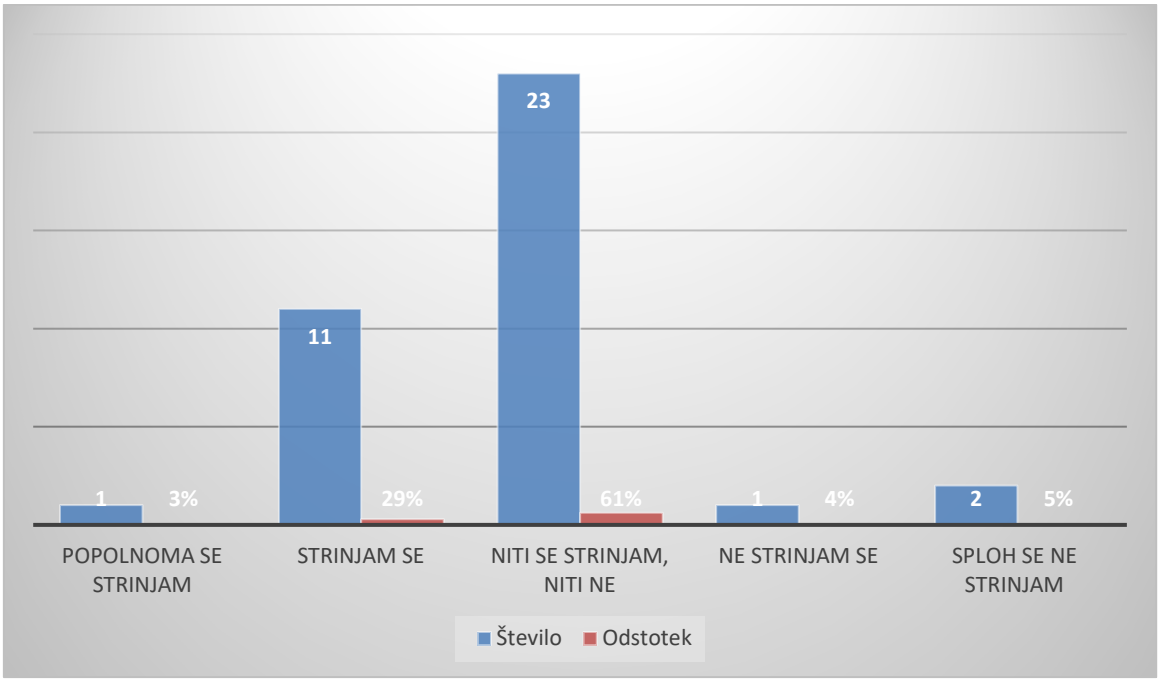
32 % nevtralnih uporabnikov verjetno ni prepričanih, ker:

- ne poznajo točnih stroškov (74 % iz prejšnjega vprašanja),
- nimajo izkušenj z obema tipoma sistemov,
- niso odgovorni za finančne odločitve.

V kombinaciji z dejstvom, da **84 % uporabnikov že uporablja lokalne sisteme**, ti rezultati **močno potrjujejo hipotezo H3 – lokalne namizne aplikacije brez mesečne naročnine so**

resnično stroškovno učinkovitejša rešitev za mala podjetja v primerjavi z oblačnimi sistemi z mesečnimi naročninami.

4.5.4 Ocena dolgoročnih stroškov mesečnih naročnin za oblačne sisteme



Graf 15: Ocena dolgoročnih stroškov za oblačne storitve

(Lastni vir, 2025)

To vprašanje prinaša **bolj nevtralne rezultate** v primerjavi s prejšnjim. **Največ uporabnikov (61 %) je nevtralnih** glede trditve, da mesečne naročnine predstavljajo prevelik strošek. To kaže, da **mnogi nimajo jasnega mnenja** o tej temi.

32 % uporabnikov (3 % + 29 %) se strinja, da so mesečne naročnine **dolgoročno prevelik strošek**, medtem ko se **8 %** (3 % + 5 %) s tem ne strinja.

Zakaj toliko nevtralnih?

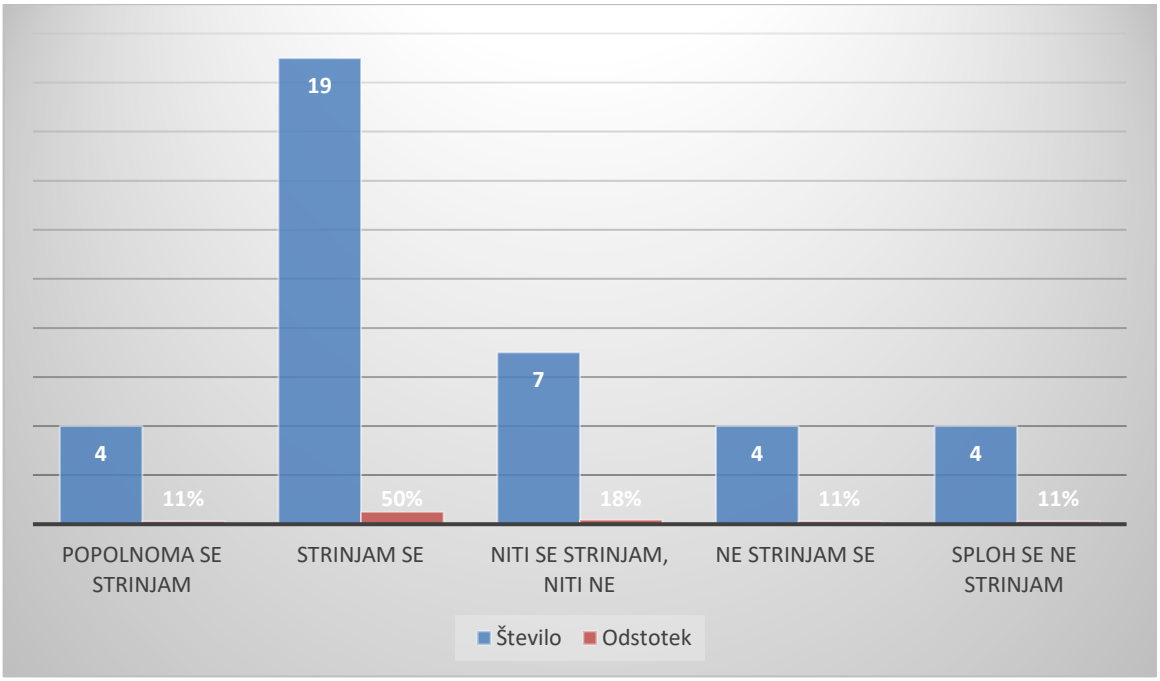
- **74 % ne ve, koliko plačujejo** (Q17).
- **84 % uporablja lokalne sisteme** (Q16) – nimajo izkušenj z naročninami.
- Mnogi niso odgovorni za finančne odločitve.

Kljub nevtralnim odgovorom pa kombinacija vseh treh vprašanj **še vedno potrjuje hipotezo H3**:

1. **84 % uporablja lokalne sisteme** (Q16),
2. **65 % meni, da so lokalne rešitve stroškovno učinkovitejše** (Q18),
3. **32 % meni, da so naročnine prevelik strošek** + **61 % je nevtralnih** (Q19).

To kaže, da uporabniki **preferirajo lokalne rešitve**, tudi če nimajo neposrednih izkušenj s stroški oblačnih alternativ.

4.5.5 Vrednost avtomatskih posodobitev in tehnične podpore



Graf 16: Pripravljenost plačevanja naročnine za avtomatske posodobitve in tehnično podporo

(Lastni vir, 2025)

To vprašanje prinaša **zanimive in pomembne rezultate**. Kar **61 % uporabnikov** (11 % + 50 %) je **pripravljenih plačevati mesečno naročnino**, če to prinaša **avtomatske posodobitve in tehnično podporo**. To kaže, da uporabniki **cenijo te dodatne storitve** in so pripravljeni zanje plačati.

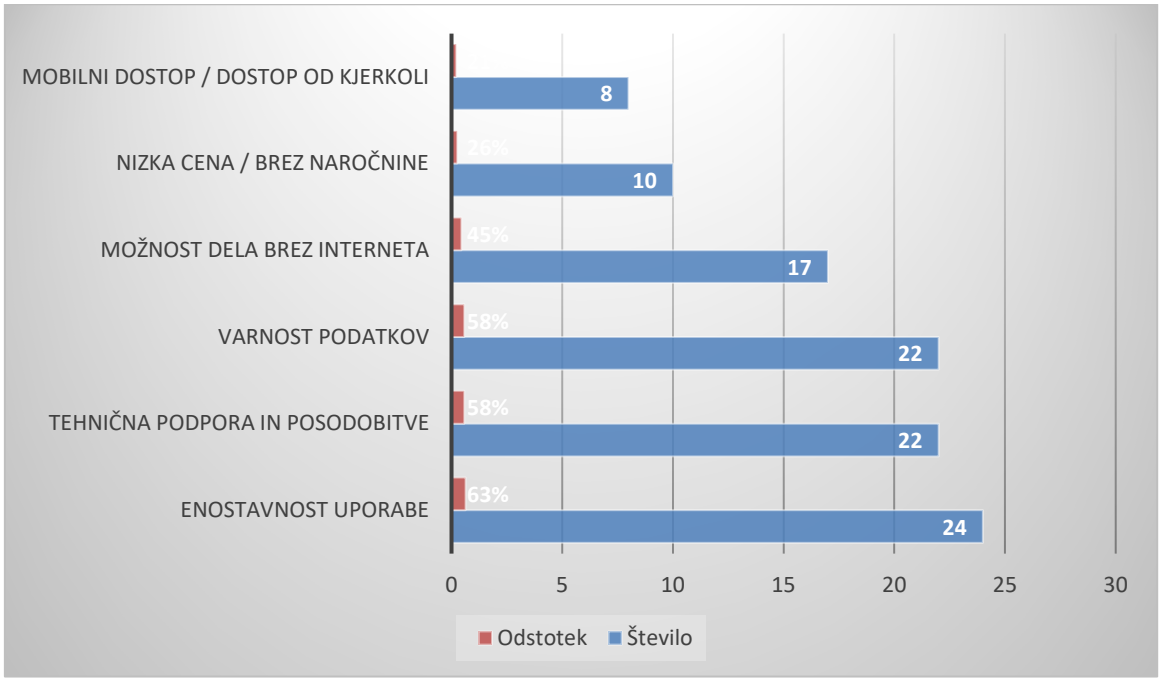
Po drugi strani pa je **22 % uporabnikov** (11 % + 11 %) jasno **proti mesečnim naročninam**, tudi če te vključujejo posodobitve in podporo. **18 % je nevtralnih**.

Ključen zaključek: ta rezultat **ne nasprotuje hipotezi H3**, ampak jo **dopolnjuje**:

- uporabniki **preferirajo lokalne rešitve brez naročnin** (Q16, Q18),
- so **pripravljeni plačevati naročnino**, če dobijo **resnično vrednost** (avtomatske posodobitve, tehnična podpora),
- problem oblačnih rešitev torej **ni naročninski model sam**, ampak **razmerje med ceno in vrednostjo**.

To pomeni, da bi lahko **hibridni modeli** (lokalna aplikacija + opcijska naročnina za podporo) predstavljali **najboljšo rešitev** za mala podjetja – imajo prednosti lokalnega sistema (enkratni strošek, delovanje brez interneta) in možnost plačevanja za dodatne storitve, če jih potrebujejo.

4.5.6 Prioritete uporabnikov pri izbiri sistema POS



Graf 17: Najpomembnejše lastnosti pri izbiri sistema POS

(Lastni vir, 2025)

To vprašanje prinaša **izjemno pomembne rezultate** za razumevanje prioritet uporabnikov. **Najpomembnejša lastnost** je **preprostost uporabe** (63 %), kar **močno potrjuje hipotezo H1** – uporabniki resnično cenijo intuitivne in preproste sisteme.

Na **drugem mestu** sta **tehnična podpora/posodobitve** in **varnost podatkov** (obe 58 %), kar kaže, da uporabniki cenijo **kakovost storitve** in **zaščito podatkov** bolj kot samo nizko ceno.

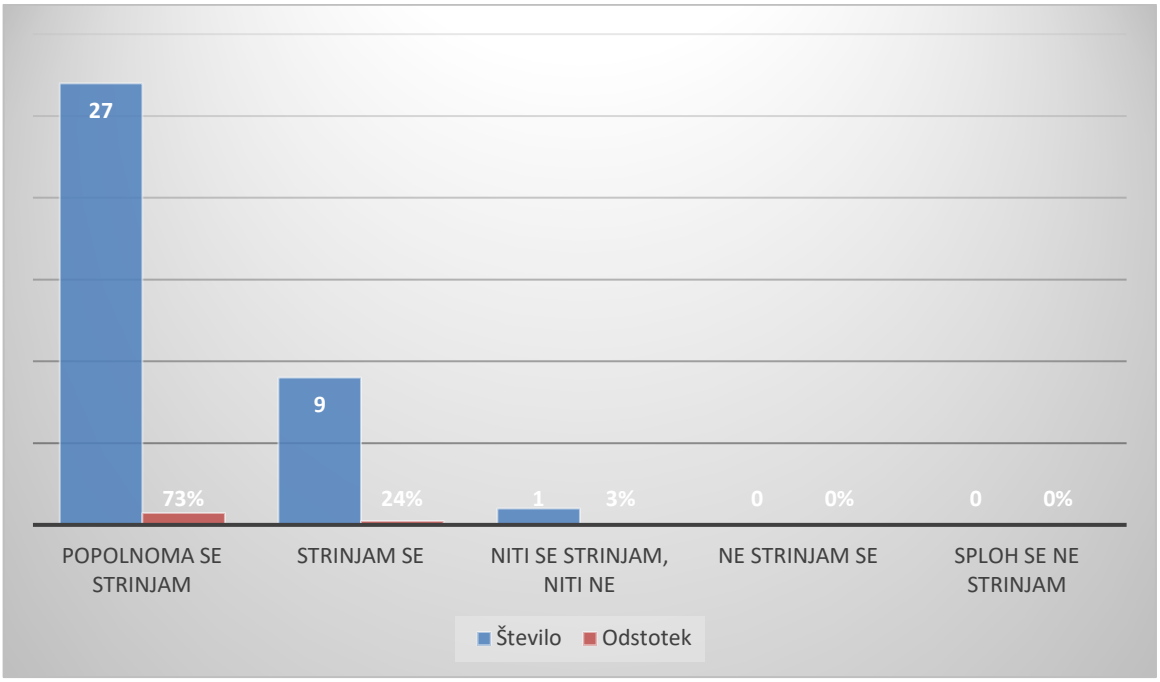
Ključni uvidi za hipotezo H3:

- **nizka cena (26 %) je šele na 5. mestu**. To pomeni, da uporabniki **ne iščejo samo najcenejše rešitve**, ampak **vrednost za denar**,
- **možnost dela brez interneta** (45 %) je pomembna, kar favorizira lokalne sisteme,
- **mobilni dostop** (21 %) je najmanj pomemben, kar pomeni, da klasične oblačne prednosti niso prioriteta.

Zaključek: uporabniki **preferirajo lokalne sisteme** (H3), ne samo zaradi cene, ampak zaradi **kombinacije lastnosti**: preprostost, delo v načinu brez internetne povezave, varnost. So pa pripravljeni plačevati za **tehnično podporo in posodobitve**, kar potrjuje rezultat iz Q20.

4.6 Varnost

4.6.1 Pomembnost varnosti podatkov



Graf 18: Pomembnost varovanja podatkov

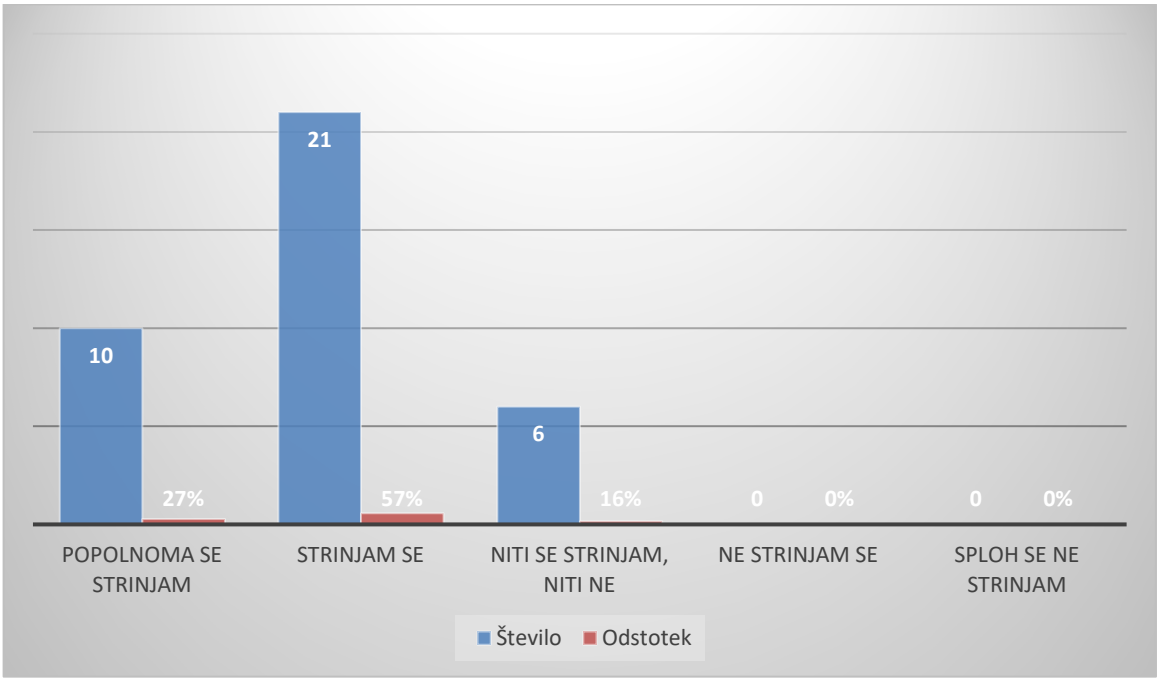
(Lastni vir, 2025)

To je **eden najjasnejših rezultatov celotne ankete**. Kar **97 % uporabnikov** (73 % + 24 %) se **strinja ali popolnoma strinja**, da je **varnost podatkov zelo pomembna**. Samo **3 % uporabnikov so nevtralni**, medtem ko **nihče** ni izbral možnosti "Ne strinjam se" ali "Sploh se ne strinjam".

To je **najvišja stopnja soglasja** med vsemi vprašanji in jasno kaže, da je **varnost podatkov absolutna prioriteta** za uporabnike sistemov POS. Ker sistemi POS obdelujejo občutljive podatke (gesla uporabnikov, finančne transakcije, včasih tudi osebne podatke strank), je ta rezultat popolnoma logičen.

Ta izjemno visoka ocena **neposredno podpira hipotezo H4** – uporabniki **ne samo, da cenijo**, ampak **zahtevajo** varnostne mehanizme v aplikacijah POS. To pomeni, da morajo razvijalci **obvezno vključiti** šifriranje gesel BCrypt, komunikacijo HTTPS/TLS, zaščito pred injekcijo SQL in druge varnostne mehanizme, ki so bili opisani v praktičnem delu diplomskega dela.

4.6.2 Zaupanje v varnostne mehanizme



Graf 19: Zaupanje v trenutni varnosti sistem

(Lastni vir, 2025)

Rezultati kažejo **visoko stopnjo zaupanja** v varnost trenutnih sistemov POS. Kar **84 % uporabnikov** (27 % + 57 %) **zaupa**, da njihov sistem ustrezno varuje občutljive podatke. Nihče ni izbral negativnih možnosti, kar pomeni, da **ni izrazitega nezaupanja** v uporabljene sisteme.

16 % nevtralnih uporabnikov verjetno ne pozna podrobnosti o varnostnih mehanizmih svojega sistema ali niso prepričani o njihovi učinkovitosti.

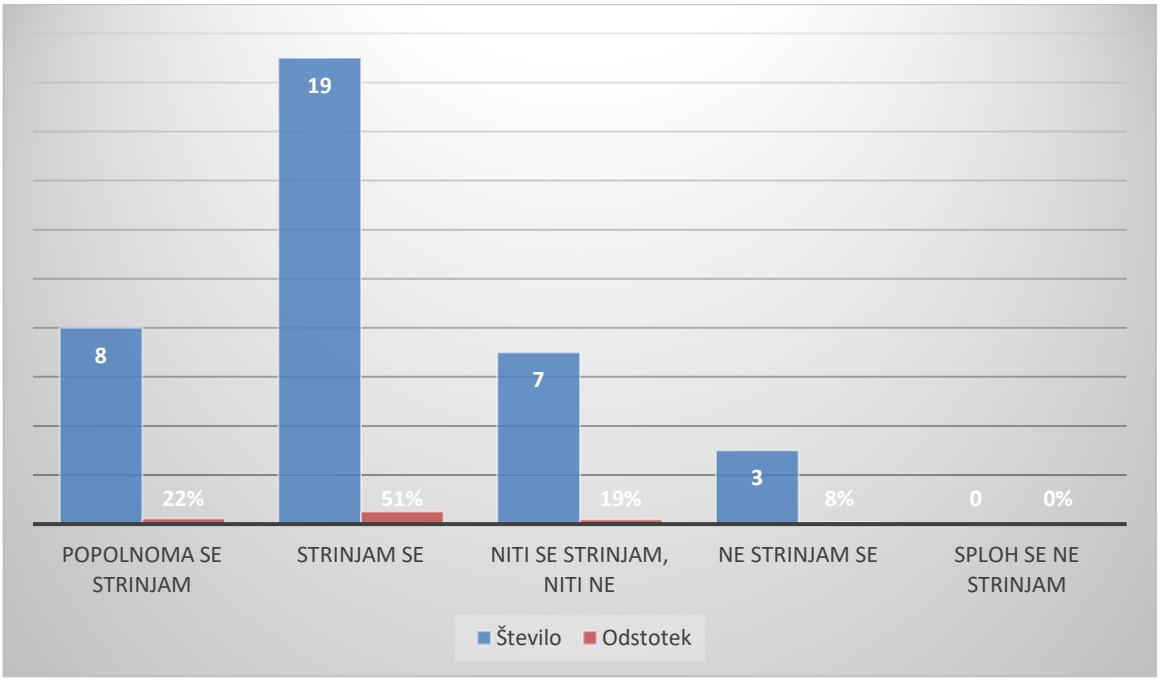
Primerjava s Q23:

- Q23: 97 % meni, da je varnost **pomembna**,
- Q24: 84 % **zaupa**, da je sistem varen.

To pomeni, da čeprav **vsi uporabniki cenijo varnost**, ne **zaupajo vsi popolnoma** svojim trenutnim sistemom. Ta 13-% razlika (97 % – 84 %) predstavlja **potencial za izboljšave** in priložnost za ponudnike, ki lahko jasno **komunicirajo svoje varnostne mehanizme** (šifriranje BCrypt, HTTPS/TLS, zaščita pred injekcijo SQL), da povečajo zaupanje uporabnikov.

Ti rezultati **podpirajo hipotezo H4** – vključitev varnostnih mehanizmov je ključna, saj uporabniki to cenijo in pričakujejo, njihovo zaupanje pa je odvisno od dejanske implementacije teh mehanizmov.

4.6.3 Vpliv varnostnih mehanizmov na izbiro sistema POS



Graf 20: Pomembnost varnostnih mehanizmov pri izbiri sistema

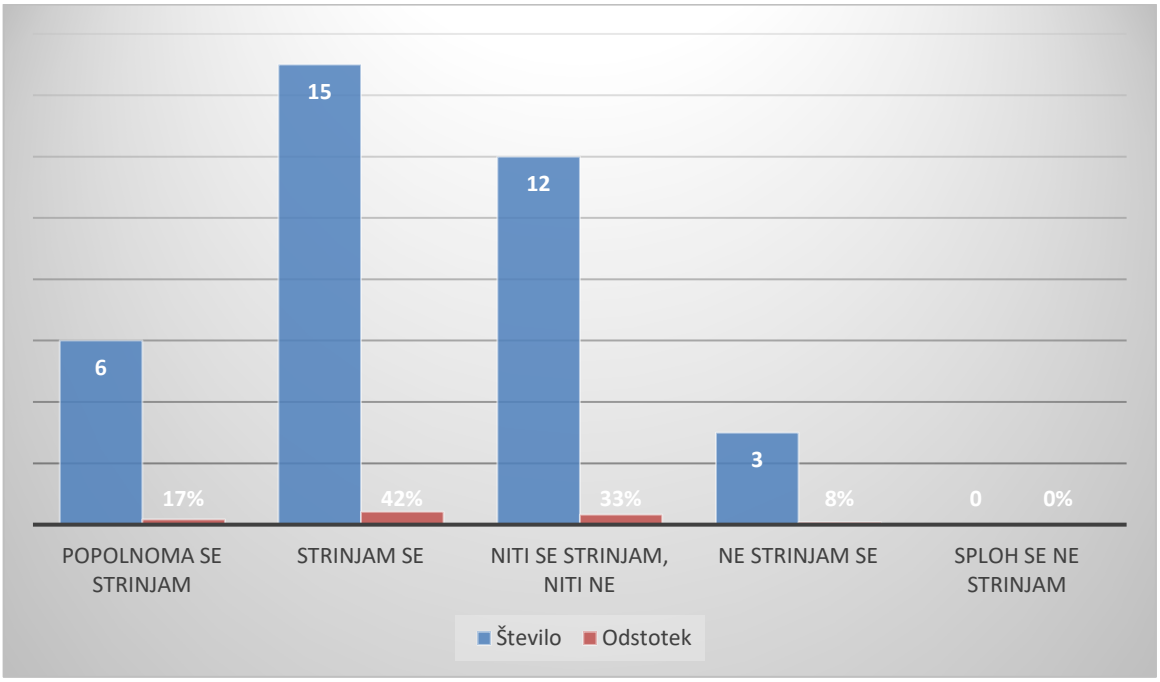
(Lastni vir, 2025)

To vprašanje prinaša **ključne rezultate za hipotezo H4**. Kar **73 % uporabnikov** (22 % + 51 %) se **strinja ali popolnoma strinja**, da **varnostni mehanizmi pomembno vplivajo na njihovo odločitev** pri izbiri sistema POS. To jasno potrjuje, da uporabniki **aktivno iščejo in cenijo** sisteme z implementiranimi varnostnimi funkcijami.

19 % nevtralnih uporabnikov verjetno ne razume tehničnih podrobnosti varnostnih mehanizmov ali pa niso neposredno odgovorni za izbiro sistema. Zanimivo je, da **se 8 % ne strinja**, kar lahko pomeni, da jim druge lastnosti (cena, preprostost) prevladajo nad varnostjo.

Ti rezultati **močno potrjujejo hipotezo H4** – vključitev varnostnih mehanizmov ne samo zmanjša možnosti zlorabe podatkov, ampak **aktivno vpliva na odločitve uporabnikov** pri izbiri sistema POS.

4.6.4 Vrednost dvofaktorske avtentikacije za zaupanje



Graf 21: Dvofaktorska avtentikacija kot dodatna varnost

(Lastni vir, 2025)

Rezultati kažejo **zmerno pozitiven odnos** do dvofaktorske avtentikacije. **59 % uporabnikov** (17 % + 42 %) se **strinja**, da bi jim 2FA dala **dodatno zaupanje v varnost** sistema. To kaže, da uporabniki **poznajo in cenijo** napredne varnostne mehanizme.

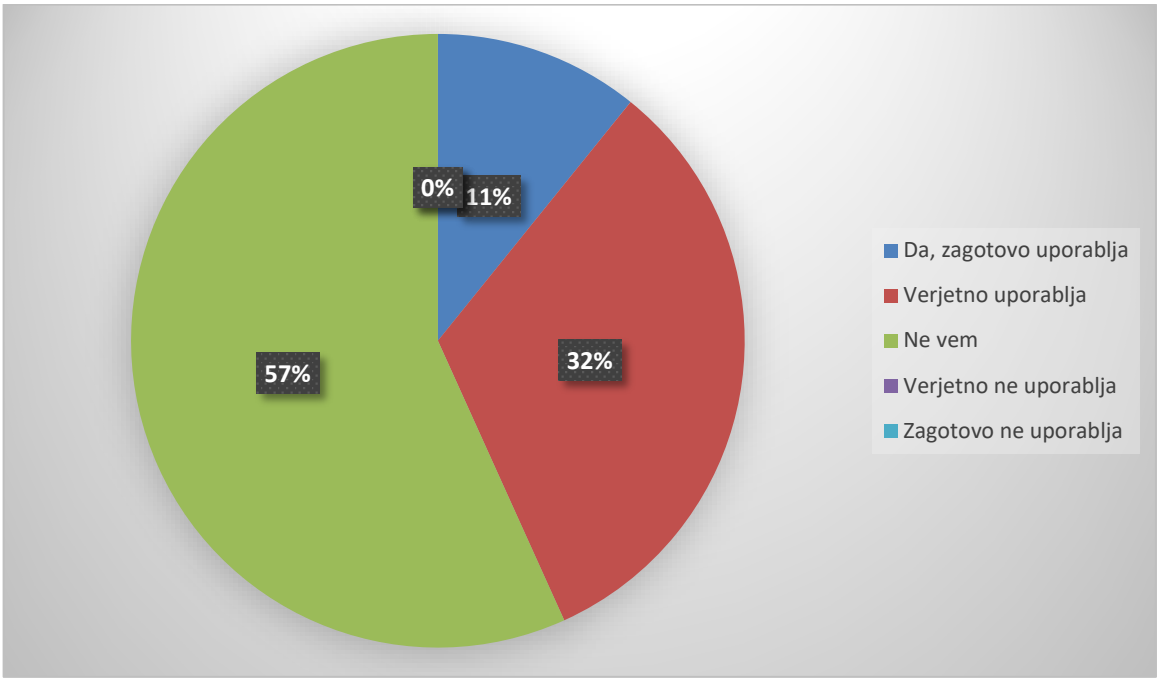
33 % nevtralnih uporabnikov je presenetljivo visok delež, kar lahko pomeni, da:

- ne poznajo dobro, kaj je 2FA ali kako deluje,
- niso prepričani, ali bi jim dodatna avtentikacija res pomagala,
- menijo, da je za sisteme POS morda pretirana.

8 % se ne strinja, kar je razumljivo – 2FA lahko upočasni prijavo in nekateri uporabniki preferirajo hitrost pred dodatno varnostjo.

Trend je jasen: **vsi cenijo varnost**, vendar je **2FA manj prioritetna** od osnovnih varnostnih mehanizmov. To je logično – **šifriranje gesel, HTTPS in zaščita dostopa** so **obvezni**, medtem ko je **2FA opcijska funkcija**, ki jo cenijo naprednejši uporabniki.

4.6.5 Ozaveščenost o šifriranju podatkov



Graf 22: Poznavanje varnostnih mehanizmov trenutnega sistema

(Lastni vir, 2025)

To je **zelo zanimiv in pomemben rezultat**. Kar **57 % uporabnikov ne ve**, ali njihov sistem POS uporablja šifriranje gesel in podatkov. Samo **11 % je prepričanih**, da sistem uporablja šifriranje, medtem ko **32 % verjetno** meni, da uporablja.

To pomeni, da čeprav **97 % uporabnikov ceni varnost** in **84 % zaupa svojemu sistemu**, **večina ne pozna tehničnih podrobnosti** o tem, kako je ta varnost implementirana.

Ključne ugotovitve

1. **Velika vrzel v poznavanju** – uporabniki ne poznajo varnostnih mehanizmov.
2. **Zaupanje brez preverjanja** – uporabniki zaupajo sistemom, ne da bi vedeli, kako delujejo.
3. **Priložnost za razvijalce** – jasna komunikacija o varnostnih funkcijah lahko poveča zaupanje.

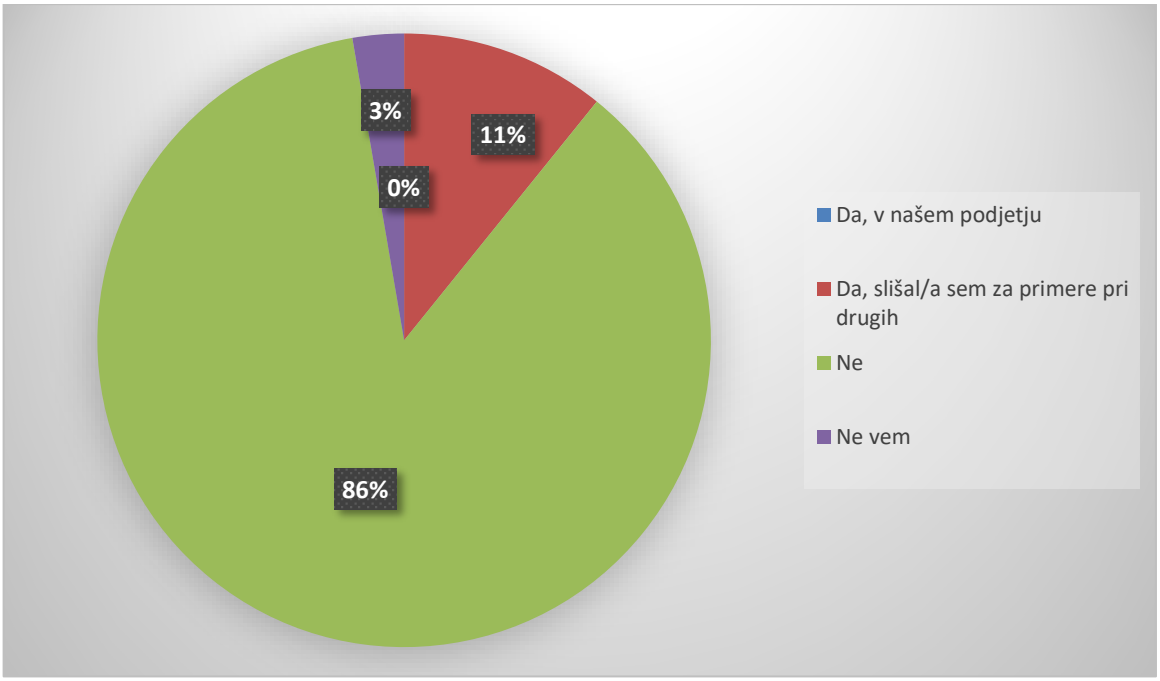
Povezava s predhodnimi rezultati:

- 73 % uporabnikov meni, da varnost vpliva na odločitev,
- 57 % uporabnikov ne ve, ali ima njihov sistem šifriranje.

To kaže, da uporabniki **želijo varnost**, ampak **ne znajo preveriti**, ali jo imajo. Razvijalci morajo **jasno komunicirati** implementirane varnostne mehanizme (BCrypt, HTTPS/TLS itd.) in jih predstaviti na razumljiv način.

Za H4: ta rezultat dodatno potrjuje pomembnost vključitve varnostnih mehanizmov – ne samo s tehničnega vidika, ampak tudi **transparentno komunikacijo** o njih.

4.6.6 Izkušnje z varnostnimi kršitvami



Graf 23: Pretekle izkušnje z varnostnimi incidenti

(Lastni vir, 2025)

To je **zelo pozitiven rezultat**! Kar **86 % uporabnikov** ni bilo priča nobeni varnostni kršitvi v sistemih POS, **nihče** pa ni doživel incidenta v lastnem podjetju. Samo **11 %** je slišalo za primere pri drugih podjetjih.

To kaže, da so **trenutni sistemi POS razmeroma varni** in da **varnostni incidenti niso pogosti** v praksi. To delno pojasnjuje tudi rezultat, kjer je 84 % uporabnikov zaupalo, da njihov sistem ustrezno varuje podatke – večina nima negativnih izkušenj.

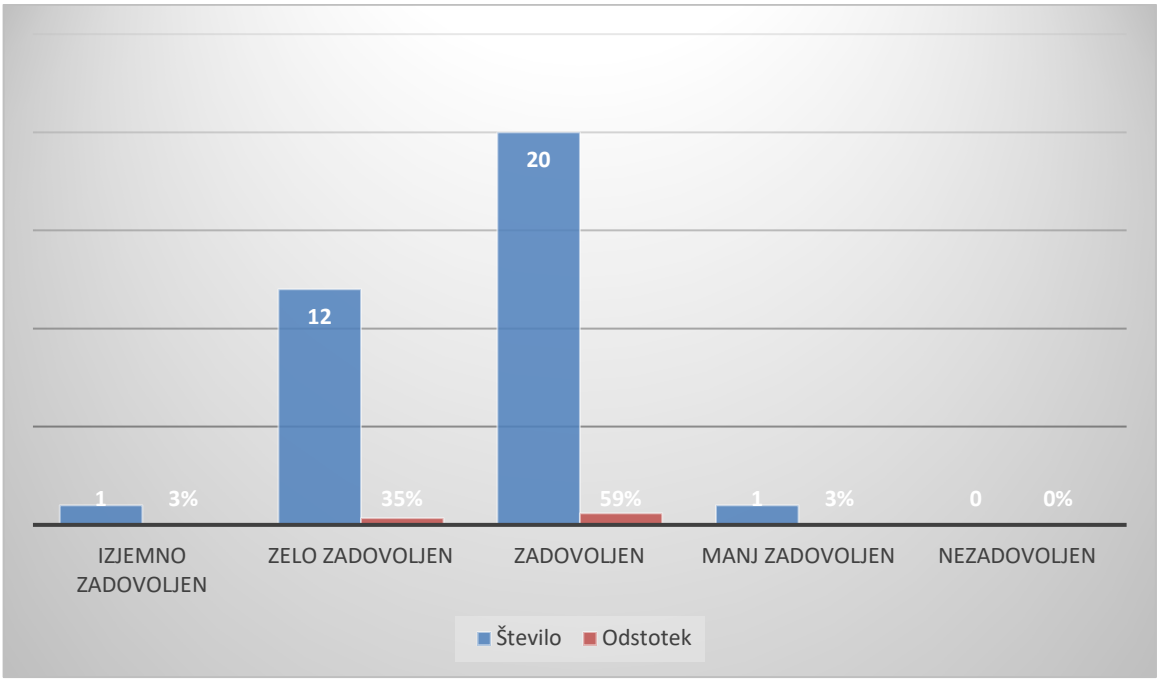
Pomembna opomba: čeprav so incidenti redki, to **ne zmanjšuje pomembnosti varnosti**. Nasprotno:

1. **nizka stopnja incidentov je rezultat dobrih varnostnih praks** (šifriranje, HTTPS, zaščita dostopa),
2. **brez varnostnih mehanizmov bi bilo incidentov več**,
3. **97 % uporabnikov še vedno ceni varnost** – previdnost je upravičena.

Zaključek za H4: nizka stopnja varnostnih incidentov **potrjuje učinkovitost obstoječih varnostnih mehanizmov** in dokazuje, da je **vključitev šifriranja BCrypt, komunikacije HTTPS/TLS in drugih varnostnih funkcij** bistvena za **preventivo** pred zlorabami. Uporabniki morda ne doživljajo incidentov prav zato, ker sistemi že vključujejo te mehanizme.

4.7 Splošno zadovoljstvo in priporočila

4.7.1 Splošno zadovoljstvo s trenutnim sistemom POS



Graf 24: Splošno zadovoljstvo s trenutnim sistemom POS

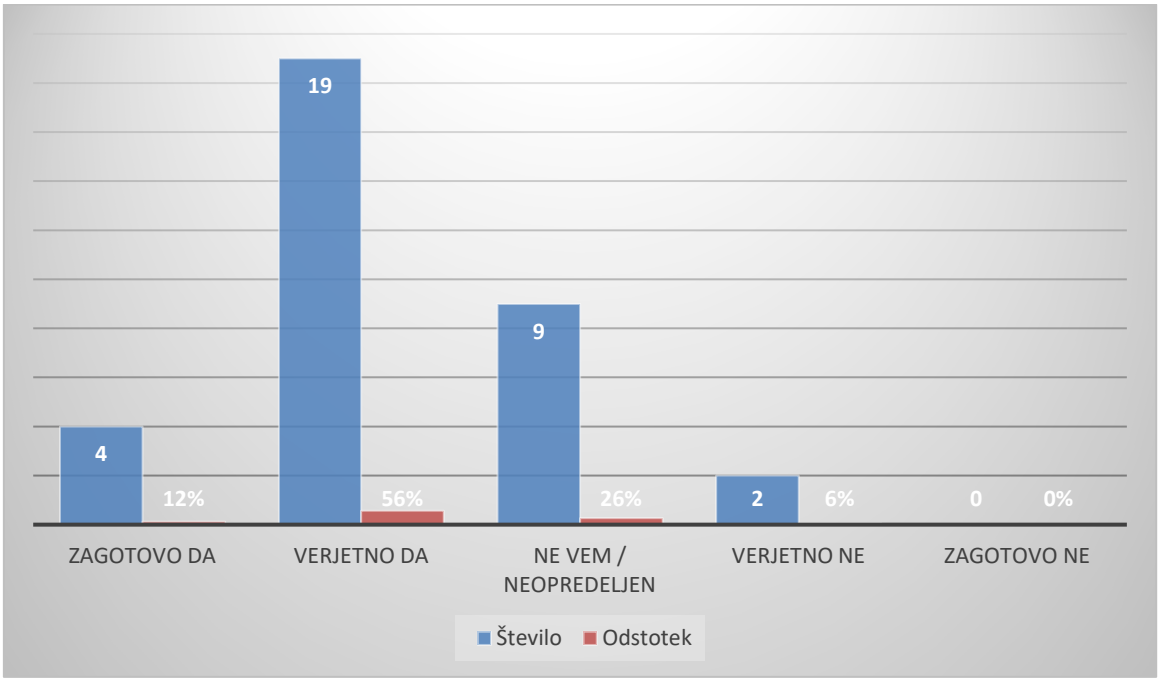
(Lastni vir, 2025)

Rezultati kažejo **visoko stopnjo splošnega zadovoljstva** s trenutnimi sistemi POS. Kar **97 % uporabnikov** (3 % + 35 % + 59 %) je **zadovoljnih ali bolj kot zadovoljnih** s svojim sistemom. Nihče ni izbral možnosti "Nezadovoljen", samo **3 % uporabnikov je manj zadovoljnih**.

Večina uporabnikov (59 %) je "Zadovoljnih", kar je solidna ocena, medtem ko je **38 % zelo ali izjemno zadovoljnih**. To kaže, da **trenutni sistemi dobro opravljajo svoje delo**, čeprav obstaja prostor za izboljšave.

To **potrjuje hipotezo H5** – uporabniki, ki so ocenili sistem kot **preprost, hiter in vizualno privlačen**, so tudi **splošno zadovoljni** z njim. Dober uporabniški vmesnik **neposredno vpliva** na splošno zadovoljstvo.

4.7.2 Pripravljenost za priporočilo sistema



Graf 25: Priporočilo sistema drugim uporabnikom

(Lastni vir, 2025)

Rezultati kažejo **pozitivno pripravljenost za priporočilo**. Kar **68 % uporabnikov** (12 % + 56 %) bi **priporočilo svoj sistem drugim**, medtem ko le **6 % verjetno ne bi**. Nihče ni izbral možnosti "Zagotovo ne".

26 % neopredeljenih je razmeroma visok delež, kar lahko pomeni, da:

- so zadovoljni, vendar ne dovolj navdušeni za aktivno priporočilo,
- ne poznajo dovolj alternativ za primerjavo,
- menijo, da vsako podjetje potrebuje drugačno rešitev.

To **potrjuje hipotezo H5** – uporabniki, ki so zadovoljni z UI/UX, preprostostjo in funkcionalnostjo, so pripravljeni **priporočiti sistem**, kar je **najboljši kazalnik uspeha** aplikacije.

4.7.3 Pozitivni vidiki trenutnih sistemov

Odgovori uporabnikov (kvalitativna analiza):

Najpogostejše omembe:

- **preprostost** (večkrat omenjeno),
- **preprostost uporabe** (večkrat omenjeno),
- **hitro se ga da naučiti,**
- **preprost vnos artikla na blagajni,**
- **uporaba,**
- **je preprost,**
- **preprosto delovanje, podpora je zelo dobra.**

Kvalitativna analiza odprtih odgovorov jasno kaže, da sta **najpogosteje omenjeni lastnosti "preprostost"** in **"preprostost uporabe"**. To **neposredno potrjuje hipotezo H1** – uporabniki resnično **najbolj cenijo intuitivne in preproste sisteme**.

Pomembna je tudi omemba **"podpora je zelo dobra"**, kar se povezuje z rezultati o pripravljenosti plačevanja za tehnično podporo.

Ta **triangulacija podatkov** (kvantitativni + kvalitativni) **močno potrjuje**, da je **preprostost uporabe ključni faktor** uspeha sistemov POS.

4.7.4 Možnosti za izboljšave

Odgovori uporabnikov (kvalitativna analiza)

Najpogostejše omembe:

- **delo brez interneta** (omenjeno večkrat),
- **hitrost** (omenjeno večkrat),

- **hitrost izstavljanja računov**,
- **možnost hitrejšega zaključka blagajne**,
- **videz**,
- **oblika**,
- **nič** (večkrat – zadovoljni, brez predlogov),
- **ne vem/ne vem, verjetno nič** (večkrat).

Kvalitativna analiza predlogov za izboljšave razkriva **tri glavne kategorije**.

1. Hitrost in učinkovitost: več uporabnikov omenja željo po **hitrejši obdelavi** (hitrost izstavljanja računov, hitrejši zaključek blagajne). To je v skladu z rezultati Q7e, kjer je hitrost dobila oceno 4,5/5 – uporabniki že cenijo hitrost, vendar si želijo **še več optimizacije**.

2. Delo brez interneta (offline mode): to je **pomemben predlog**, ki se povezuje z:

- 52 % uporabnikov ne more nemoteno delovati ob izpadu FURS (potrebujejo način brez internetne povezave),
- H2: zanesljiva integracija FURS je ključna. **Priporočilo**: implementacija načina **brez internetne povezave** z odloženim pošiljanjem na FURS bi pomembno izboljšala uporabniško izkušnjo.

3. Videz: nekateri omenjajo **videz/obliko**, kar potrjuje pomembnost UI/UX (Q7f: 4,1/5, Q21: videz pomemben).

Pomembno: veliko uporabnikov je odgovorilo **"Nič"** ali **"Ne vem"**, kar kaže na **visoko zadovoljstvo** s trenutnimi sistemi (potrjuje Q31: 97 % zadovoljnih).

5 SKLEP

Diplomsko delo je obravnavalo razvoj sodobne namizne aplikacije za blagajniško poslovanje z integrirano povezavo s sistemom FURS za davčno potrjevanje računov. V teoretičnem delu so bili predstavljeni analiza globalnega trga sistemov POS, zakonodajni okvir za davčno potrjevanje, varnostni standardi in načela uporabniške izkušnje. Praktični del je opisal celovit razvoj aplikacije z uporabo tehnologij .NET 8.0, Avalonia UI Framework in Microsoft SQL Server, s poudarkom na implementaciji varnostnih mehanizmov in pristopu Clean Architecture.

Doseganje zastavljenih ciljev

Vsi zastavljeni cilji diplomskega dela so bili uspešno doseženi. Razvita je bila **funkcionalna namizna aplikacija** za blagajniško poslovanje, ki vključuje avtentikacijo uporabnikov, procesiranje plačil in simulacijo davčnega potrjevanja preko okolja FURS sandbox. Implementirani so bili **varnostni mehanizmi**, vključno s hashiranjem gesel BCrypt, komunikacijo HTTPS/TLS, zaščito pred injekcijo SQL in skladnostjo z GDPR pri minimizaciji osebnih podatkov.

Za evalvacijo **potreb in pričakovanj uporabnikov** sistemov POS ter validacijo zastavljenih hipotez je bila izvedena **uporabniška anketa** med 47 potencialnimi uporabniki, večinoma blagajniki in prodajalci iz različnih sektorjev. Anketa je bila osredotočena na **splošne izkušnje z obstoječimi sistemi POS**, njihove prioritete, pričakovanja glede funkcionalnosti in razumevanje ključnih zahtev, ki jih mora izpolnjevati sodobna blagajniška aplikacija.

Validacija raziskovalnih hipotez

H1: Uporabniki pričakujejo hitro in intuitivno blagajniško rešitev.

Hipoteza H1 je bila **v celoti potrjena**. Rezultati ankete jasno kažejo, da je **preprostost uporabe absolutna prioriteta** – kar 63 % anketirancev jo je navedlo kot najpomembnejšo lastnost pri izbiri sistema POS. Uporabniki visoko cenijo intuitivnost (ocena 4,0/5) in hitrost delovanja (4,5/5), pri čemer **ne pričakujejo potrebe po obsežnem usposabljanju** (ocena 1,9/5). Kvalitativna analiza dodatno potrjuje, da je "preprostost" najpogosteje omenjena pozitivna

lastnost. Te ugotovitve so vodile razvoj aplikacije, kjer je bil uporabniški vmesnik zasnovan po načelih čistega dizajna, jasne navigacije in minimaliziranja števila korakov za osnovne operacije.

H2: Zanesljiva integracija z davčnim potrjevanjem FURS je ključna.

Hipoteza H2 je bila **v celoti potrjena**. Kar **82 % uporabnikov** potrjuje, da je zanesljiva integracija FURS ključna za nemoteno delovanje, **90 %** pa ceni avtomatizacijo davčnega potrjevanja. Več kot polovica uporabnikov ne more nemoteno delovati ob izpadu sistema FURS, kar potrjuje kritičnost te funkcionalnosti. V razviti aplikaciji je bila implementirana simulacija integracije FURS preko okolja sandbox z generiranjem ZOI (Zaščitna Oznaka Izdajatelja) in simulacijo EOR (Enkratna Oznaka Računa), kar predstavlja tehnično podlago za produkcijsko implementacijo.

H3: Lokalne aplikacije so stroškovno učinkovitejše za mala podjetja.

Hipoteza H3 je bila **potrjena**. Trenutno **84 % uporabnikov** uporablja lokalne namizne aplikacije, kar kaže na prevlado tega modela v praksi. **65 % anketirancev** se strinja, da so lokalne rešitve brez mesečne naročnine stroškovno učinkovitejše za majhna podjetja. Pomembna je ugotovitev, da nizka cena sama po sebi ni prioriteta (26 %), temveč uporabniki iščejo **optimalno razmerje med ceno in vrednostjo**. Razvita aplikacija predstavlja stroškovno učinkovito alternativo dragim komercialnim sistemom, še posebej primerno za mala podjetja, samostojne podjetnike in gostinske lokale.

H4: Varnostni mehanizmi zmanjšajo možnost zlorabe podatkov.

Hipoteza H4 je bila **v celoti potrjena**. Varnost podatkov je **najvišje ocenjena prioriteta** – kar **97 % uporabnikov** meni, da je zelo pomembna. **73 %** uporabnikov potrjuje, da varnostni mehanizmi vplivajo na njihovo izbiro sistema POS. Nizka stopnja varnostnih incidentov v praksi (86 % uporabnikov ni bilo priča kršitvam) potrjuje učinkovitost varnostnih praks. V razviti aplikaciji so bili implementirani vsi ključni varnostni mehanizmi: BCrypt hashing za gesla, HTTPS/TLS za komunikacijo, zaščita pred injekcijo SQL preko parametrizacije Entity

Framework Core, skladnost z GDPR z minimizacijo osebnih podatkov in role-based avtorizacija.

H5: Dober UI/UX vpliva na zadovoljstvo in zaupanje.

Hipoteza H5 je bila **v celoti potrjena**. Uporabniki, ki visoko ocenjujejo preprostost (4,0/5), hitrost (4,5/5) in videz (4,1/5), so tudi **splošno zelo zadovoljni** s svojimi sistemi (97 % zadovoljnih). **68 %** uporabnikov bi priporočilo svoj sistem drugim, kar je najboljši kazalnik uspeha. Pri razvoju aplikacije je bil uporabniški vmesnik zasnovan po arhitekturnem vzorcu MVVM z Avalonia UI Framework, kar omogoča čist, pregleden in odziven vmesnik z jasno strukturiranimi funkcionalnostmi.

Ključne ugotovitve raziskave

Raziskava uporabniških potreb in pričakovanj je razkrila več pomembnih ugotovitev, ki so neposredno vplivale na razvoj aplikacije.

Prioritete uporabnikov: preprostost uporabe (63 %), tehnična podpora (58 %), varnost podatkov (58 %) in možnost dela brez interneta (45 %) so najpomembnejši dejavniki. Nizka cena je šele na petem mestu (26 %), kar potrjuje, da uporabniki iščejo **vrednost za denar**.

Hitrost delovanja: 81 % uporabnikov izda račun v manj kot 30 sekundah, kar potrjuje, da je **optimizacija hitrosti** kritična za uspeh sistemov POS.

Potreba po funkcionalnosti brez internetne povezave: več kot polovica uporabnikov ne more nemoteno delovati ob izpadu sistema FURS, kar kaže na potrebo po implementaciji načina brez internetne povezave.

Vrzel v poznavanju varnosti: čeprav 97 % uporabnikov ceni varnost, 57 % uporabnikov ne ve, ali njihov sistem uporablja šifriranje, kar kaže na pomembnost **transparentne komunikacije** varnostnih funkcionalnosti.

Diplomsko delo je uspešno doseglo vse zastavljene cilje in **potrdilo vseh pet raziskovalnih hipotez**. Razvita aplikacija predstavlja funkcionalno rešitev, ki upošteva ključne potrebe uporabnikov: **preprostost, hitrost in varnost**. Implementirani varnostni mehanizmi sledijo mednarodnim standardom (ISO/IEC 27001, GDPR), arhitekturni pristop (Clean Architecture, MVVM) pa omogoča preprosto vzdrževanje in razširljivost.

Raziskava uporabniških potreb je pokazala, da je **ključ do uspeha aplikacij POS** v odličnem uporabniškem vmesniku in zanesljivem delovanju, ne pa zgolj v nizki ceni. Razvita rešitev dokazuje, da je možno **z omejenimi viri** ustvariti aplikacijo, ki sledi sodobnim standardom in izpolnjuje zakonodajne zahteve.

Z implementacijo predlaganih izboljšav, še posebej načina brez internetne povezave in produkcijske integracije FURS, bi aplikacija lahko postala **konkurenčna alternativa** obstoječim komercialnim rešitvam na slovenskem trgu, predvsem za segment malih podjetij in samostojnih podjetnikov.

6 VIRI IN LITERATURA

Avalonia. (2025). Pridobljeno iz https://docs.avaloniaui.net/docs/overview/what-is-avalonia

Covington & Burling. (2024). Pridobljeno iz https://www.cov.com/en/news-and-insights/insights/2015/12/the-new-eu-data-protection-law

Crown Information Management. (25. July 2024). Pridobljeno iz https://www.crownrms.com/insights/gdpr-is-six-years-old-what-is-changing-in-2024/

ECSO. (2024). Pridobljeno iz https://ecs-org.eu/activities/nis2-directive-transposition-tracker/

European Commission. (2022). Pridobljeno iz https://www.nis-2-directive.com/

European Union Agency for Fundamental Rights. (2024).

FURS. (26. 8. 2025). *Davčno potrjevanje računov.*

ISACA. (9. May 2023). Pridobljeno iz https://www.isaca.org/resources/news-and-trends/industry-news/2023/what-the-post-pandemic-future-holds-for-it-auditors

ISDDecisions. (2024). Pridobljeno iz https://www.isdecisions.com/en/blog/compliance/guide-to-tisax-certification

ISO. (2022). Pridobljeno iz https://secureframe.com/blog/iso-27001-2022

OKUMOKU-EVRORO, O. (2025). *Academia.edu*. Pridobljeno iz Development of an Enhanced Point of Sales System for Retail Business in Developing Countries

Računalniške novice. (2024).

Research, V. M. (2024). *Cloud POS Market Size and Forecast*. Pridobljeno iz https://www.verifiedmarketresearch.com/product/cloud-pos-market/.

Sebastina Nkechi Okofu, C. A.-E. (March 2025). *ResearchGate*. Pridobljeno iz ResearchGate: https://www.researchgate.net/publication/390329662_Development_of_an_Enhanced_Point_of_Sales_System_for_Retail_Business_in_Developing_Countries

Secureframe. (2024). Pridobljeno iz https://secureframe.com/en-us/hub/iso-27001

SI-CERT. (8. March 2024). Pridobljeno iz https://www.cert.si/zinfv-1/

Skadden. (2024). Pridobljeno iz https://www.skadden.com/insights/publications/2024/10/navigating-the-new-cybersecurity-landscape

Subhiyakto & Astuti. (2020).

TÜV Rheinland. (2024). Pridobljeno iz https://www.tuv.com/world/en/tisax.html

UpGuard. (2024). Pridobljeno iz https://www.upguard.com/resources

varnost, U. V. (4. June 2025). *Republika Slovenija*. Pridobljeno iz https://pisrs.si/pregledPredpisa?id=ZAKO8934

Wavestone. (2024). Pridobljeno iz https://www.wavestone.com/en/insight/nis-2-european-countries-transposing-directive/