Agentni sistem RAG za podporo zaposlenim v večjih podjetjih

Rok Hladin

Telekom Slovenije, d. d., Ljubljana, Slovenija
rok.hladin@telekom.si

Predstavljamo zasnovo, izvedbo in vrednotenje agentnega sistema RAG, ki zaposlenim v velikem telekomunikacijskem podjetju omogoča pogovorno iskanje informacij po bazi znanja, zgrajeni iz vsebin javne spletne strani. Sistem sestavljata cevovod za zajem podatkov, ki spletne vsebine pretvarja v vektorske vložitve v bazi Azure AI Search, in agentna aplikacija, zgrajena z ogrodjem LlamaIndex, ki po vzorcu ReAct sama odloča o hibridnem iskanju ter vsak odgovor podpre z navedbo virov. Poleg oblačnega jezikovnega modela sistem prek prehoda Bifrost za umetno inteligenco uporablja lokalno gostujoče modele, pri katerih podatki ne zapuščajo infrastrukture podjetja. Na naboru 105 realnih vprašanj testnih uporabnikov delež uspešnih odgovorov po širši opredelitvi, ki šteje tudi delno pravilne in upravičeno zavrnjene odgovore, znaša 94,3 %, delež v celoti pravilnih pa 67,6 %; prvi del odgovora uporabnik običajno prejme po približno petih sekundah. Anketa med testnimi uporabniki potrjuje prihranek časa, kot glavna izziva pa izpostavlja pokritost baze znanja in zaupanje v odgovore.

Ključne besede:
agentni sistemi
s poizvedovanjem obogateno generiranje (RAG)
veliki jezikovni modeli
vektorsko iskanje
LlamaIndex

Univerzitetna založba Univerze v Mariboru

ISBN 978-961-299-188-3
DOI https://doi.org/10.18690/um.feri.7.2026.5

1 Uvod

Zaposleni v večjih podjetjih se pri vsakodnevnem delu opirajo na obsežen in nenehno spreminjajoč se nabor informacij o produktih, storitvah, postopkih in pogodbenih pogojih. V Telekomu Slovenije so te informacije razpršene po več virih, od javne spletne strani do internih navodil in dokumentov, iskanje po njih pa je ročno: zaposleni mora sam presoditi, kateri vir informacijo vsebuje, jo v njem poiskati in preveriti njeno ažurnost. Za nekatere skupine zaposlenih je tak način dela posebej obremenjujoč. Zaposleni v klicnem centru potrebuje točen odgovor med telefonskim pogovorom z uporabnikom, prodajalec na prodajnem mestu pa mora pogoje paketa preveriti, medtem ko uporabnik čaka.

Veliki jezikovni modeli [1] so pokazali, da je dostop do informacij prek pogovora v naravnem jeziku za uporabnike izjemno učinkovit, vendar njihova neposredna uporaba v poslovnem okolju trči ob dve omejitvi: nagnjenost k halucinacijam ter znanje, omejeno na učne podatke z določenim časovnim presekom. Obe omejitvi rešuje s poizvedovanjem obogateno generiranje (angl. *Retrieval-Augmented Generation*, RAG) [2], pri katerem model pred generiranjem odgovora prejme relevantne odlomke iz zunanje baze znanja. Agentni sistemi ta pristop nadgradijo tako, da jezikovnemu modelu prepustijo odločanje o poteku reševanja naloge, na primer o tem, ali in kolikokrat bo iskal po bazi znanja ter s kakšno poizvedbo [3].

V prispevku predstavljamo izkušnje z zasnovo, izvedbo in vrednotenjem takega sistema v realnem okolju Telekoma Slovenije. Tehnična izvedba pri tem ni sama sebi namen; osrednje vprašanje je, ali tak sistem zaposlenim pri delu dejansko pomaga. Prispevek zato poleg arhitekture in izvedbenih odločitev predstavlja tudi rezultate vrednotenja na realnih vprašanjih testnih uporabnikov, meritve odzivnih časov in povratne informacije uporabnikov, ki so sistem uporabljali pri svojem delu.

2 Od klasičnega RAG do agentnega sistema

Tipičen sistem RAG deluje v treh korakih: dokumenti iz baze znanja se razdelijo na odlomke, pretvorijo v vektorske vložitve in shranijo v vektorsko podatkovno bazo; ob uporabnikovi poizvedbi se iz baze pridobijo pomensko najpodobnejši odlomki; ti se skupaj s poizvedbo vključijo v poziv jezikovnemu modelu, ki generira odgovor [2]. Odgovori so tako utemeljeni na dejanskih virih, bazo znanja pa je mogoče posodabljati neodvisno od modela.

Klasični RAG sledi fiksnemu zaporedju: ena poizvedba, eno iskanje, eno generiranje, zato odpove pri sestavljenih vprašanjih, ki zahtevajo informacije iz več virov, pri nejasnih vprašanjih, ki bi jih bilo treba preoblikovati, in pri vprašanjih, ki iskanja sploh ne potrebujejo. Agentni sistem te omejitve rešuje tako, da odločitve o poteku (ali, kolikokrat in s kakšno poizvedbo iskati) prepusti modelu; tok izvajanja s tem ni več vnaprej določen, temveč odvisen od sprotnih odločitev modela. Med agentnimi vzorci je za obravnavani sistem osrednji vzorec ReAct [3], ki združuje sklepanje in izvajanje dejanj v iterativni zanki: model izmenično oblikuje sklepni korak, izvede dejanje (na primer klic orodja za iskanje) in prejme rezultat dejanja, cikel pa se ponavlja, dokler model ne oceni, da ima dovolj informacij za odgovor. Sorodna vzorca Self-RAG [4] in CRAG [5] dodajata samorefleksijo in mehanizme za izboljševanje kakovosti pridobljenega konteksta.

Za izvedbo smo primerjali štiri zrela odprtokodna ogrodja: LangChain [6], LlamaIndex [7], Haystack 2.x [8] in Semantic Kernel [9]. Šlo je za interno kvalitativno presojo po vnaprej določenih merilih: integracije s ciljnimi storitvami (Azure OpenAI, lokalni modeli, vektorske shrambe), podpora zajemu in pripravi dokumentov, funkcionalnosti RAG (hibridno iskanje, ponovno razvrščanje, agentni delovni tokovi), vzdrževanost in licenca. Izbrali smo LlamaIndex, ker je obravnavani sistem predvsem podatkovno in dokumentno usmerjen: ogrodje je med primerjanimi možnostmi ponujalo najboljšo podporo za dokumentne bralnike, transformacije, metapodatke in indeksiranje, širok nabor konektorjev za ciljne shrambe ter podporo agentnim vzorcem. Njegova omejitev je, da na višjih ravneh abstrakcije včasih skrije del izvedbene logike, zato je za produkcijsko rabo pomembno dosledno sledenje izvajanja.

3 Zahteve in kontekst uporabe

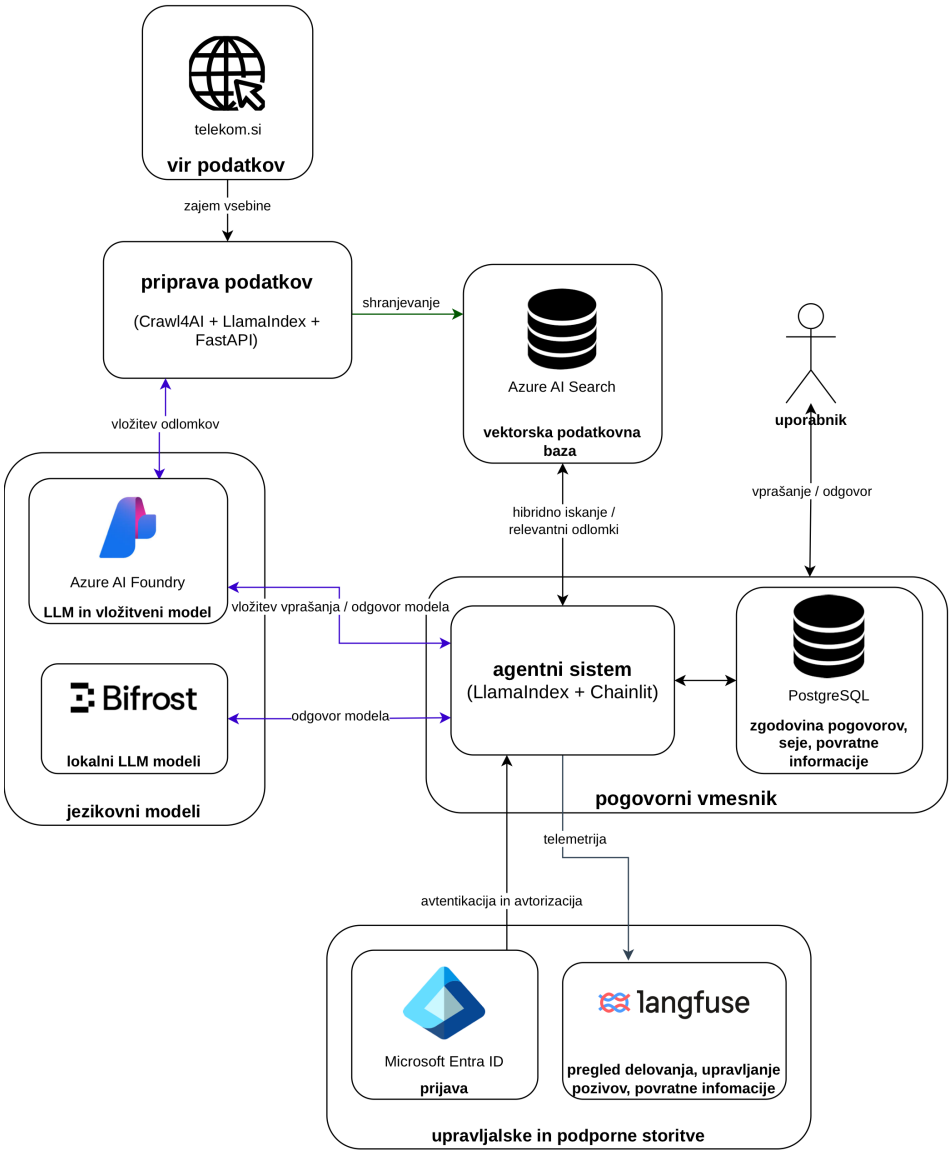
Primarne uporabniške skupine sistema so zaposleni v klicnem centru, prodajnih centrih in zalednih službah ter zunanji prodajni partnerji. Skupine se razlikujejo po obsegu potrebnih informacij in načinu dela: zaposleni v klicnem centru potrebujejo hitre odgovore med telefonskim pogovorom, zaledne službe pa podrobnejše postopkovne informacije.

Iz analize dela uporabniških skupin smo izpeljali ključne funkcionalne zahteve: odgovarjanje na vprašanja v slovenskem jeziku na podlagi vsebin iz baz znanja, obvezno navedbo vira v vsakem odgovoru, prijavo prek Microsoft Entra ID, beleženje zgodovine pogovorov, telemetrijo za spremljanje delovanja, pretočno prikazovanje odgovora med generiranjem ter jasno sporočanje omejitev, kadar sistem relevantnih vsebin ne najde. Zadnji dve zahtevi skupaj gradita zaupanje v sistem: odgovori so preverljivi prek virov, ob manjkajočih informacijah pa sistem omejitev prizna, namesto da bi vrnil nezanesljiv odgovor. Med nefunkcionalnimi zahtevami so podpora do 50 hkratnih in približno 200 dnevnih uporabnikov ter hramba vseh podatkov v Evropski uniji.

Primarni vir podatkov v prvi fazi je javno spletno mesto telekom.si s približno 3.000 stranmi, ki vsebuje informacije o produktih, storitvah, cenah in pogojih. Kot razširitev je načrtovana vključitev interne baze znanja Telekoma Slovenije, ki temelji na istem sistemu za upravljanje vsebine (angl. *content management system*, CMS) in zato dopušča enak pristop k zajemu; baza v času pisanja še ni pripravljena za uporabo, dostop do njenih vsebin pa bo pogojen z uporabniško skupino. Kot merili uspešnosti sistema smo opredelili delež uspešno odgovorjenih vprašanj in zadovoljstvo uporabnikov.

4 Arhitektura in implementacija

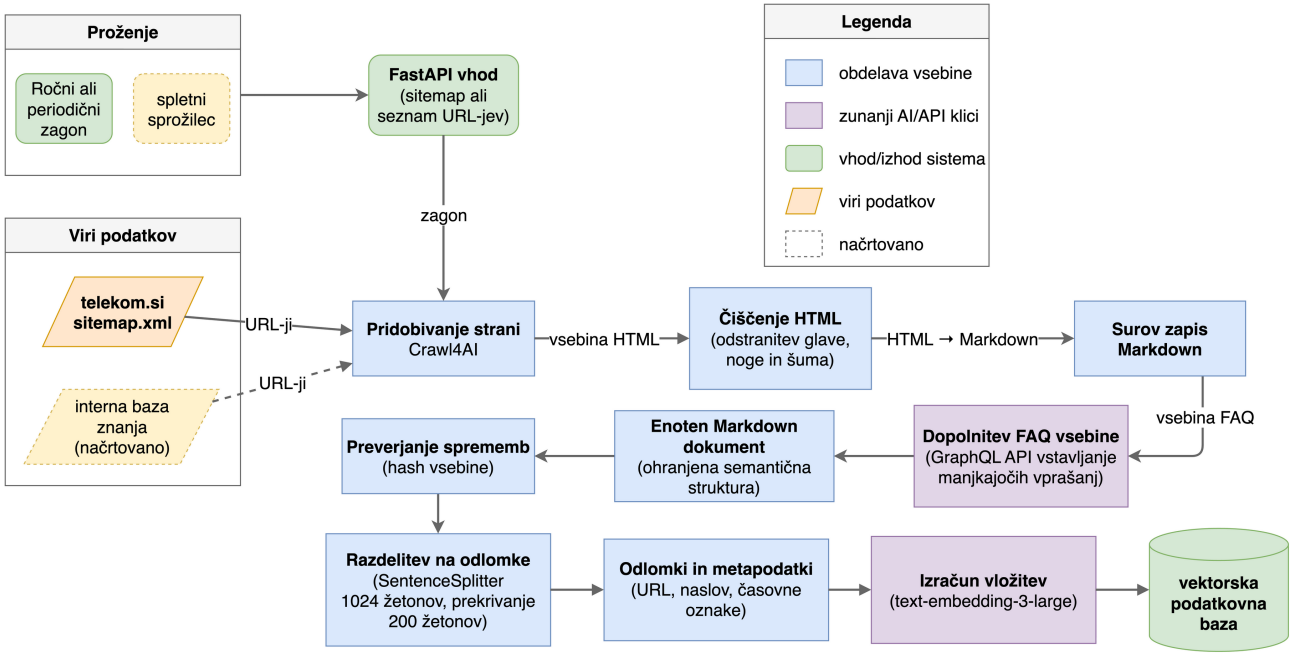
Arhitektura sistema je razdeljena na dva podsistema, ki sta povezana prek vektorske baze: cevovod za zajem podatkov, ki spletne vsebine pridobiva, obdeluje in zapisuje v vektorsko bazo, ter agentno pogovorno aplikacijo, ki ob uporabnikovem vprašanju išče po vektorski bazi in z jezikovnim modelom oblikuje odgovor. Taka delitev omogoča, da se osveževanje znanja in odgovarjanje izvajata neodvisno in z različnima frekvencama. Vse lastne komponente (cevovod, agentna aplikacija in baza PostgreSQL) tečejo kot ločeni »podi« na platformi OpenShift znotraj infrastrukture podjetja; Azure AI Search, Azure AI Foundry, Microsoft Entra ID in Langfuse so zunanje storitve. Celotno arhitekturo prikazuje slika 1.



Slika 1: Arhitektura agentnega sistema za podporo zaposlenim.

4.1. Cevovod za zajem in obdelavo podatkov

Cevovod za zajem podatkov je samostojna aplikacija, ki spletne vsebine pretvarja v obliko, primerno za semantično iskanje; njegov potek prikazuje slika 2. Seznam URL-jev pridobi iz datoteke sitemap.xml, ki v praksi predstavlja kanonični seznam objavljenih strani. Strani HTML prenese in pretvori v zapis Markdown s knjižnico Crawl4AI [10], ki je namenjena prav pripravi spletnih vsebin za jezikovne modele. Zapis Markdown ohranja semantično strukturo dokumenta (naslove, sezname, povezave) in je hkrati dovolj zgoščen za vključitev v kontekst jezikovnega modela.



Slika 2: Cevovod za zajem in obdelavo podatkov.

Surov zapis Markdown ni neposredno primeren za vektorsko iskanje, zato sledi korak čiščenja in obogatitve. Glavo in nogo strani, ki vsebujeta predvsem navigacijske elemente in se ponavljata na vseh straneh, odstranimo, saj v vektorski bazi povzročata šum. Posebno obravnavo so zahtevali razdelki s pogosto zastavljenimi vprašanji: ker so v sistemu CMS vprašanja in odgovori shranjeni v ločenih strukturah, pretvorba v Markdown zajame le odgovore, vprašanja pa izgubi. Manjkajoča vprašanja zato pridobimo neposredno prek vmesnika GraphQL, ki ga izpostavlja sistem CMS, in jih programsko vstavimo nad pripadajoči odgovor, s čimer par vprašanje-odgovor ostane povezan v istem odlomku in ga je mogoče najti prek poizvedb v naravnem jeziku.

Pripravljene dokumente razdelimo na odlomke z razdelilnikom *SentenceSplitter* iz ogrodja LlamaIndex z velikostjo 1024 žetonov in prekrivanjem 200 žetonov; meje odlomkov so poravnane s stavčnimi mejami. Velikost odlomka določa kompromis med natančnostjo iskanja in celovitostjo konteksta: pri manjših odlomkih se vsebinsko zaokrožene enote, kot so navodila po korakih, razbijejo na več delov, pri večjih pa vložitev postane manj razločevalna. Izbrana velikost praviloma v celoti zajame posamezen razdelek o storitvi, paketu ali postopku, vključno s pari vprašanje–odgovor. Vložitve odlomkov izračunamo z modelom *text-embedding-3-large* prek storitve Azure AI Foundry, ki podpira tudi neangleška besedila, in jih z operacijo *upsert* zapišemo v indeks Azure AI Search, kar omogoča ponovne zagone cevovoda brez ustvarjanja dvojnikov.

Cevovod je zapakiran v aplikacijo FastAPI in izpostavlja dve operaciji: celovit zagon nad celotno datoteko sitemap.xml in zagon nad eksplicitnim seznamom URL-jev za ciljano osvežitev. V trenutni postavitvi se proži ročno, ista zasnova pa omogoča tudi dogodkovno proženje, pri katerem sistem CMS ob spremembi vsebine prek spletnega sprožilca (angl. *webhook*) pokliče operacijo za ciljano osvežitev, tako da vektorska baza skoraj sproti sledi spremembam vsebin.

4.2. Vektorska baza in hibridno iskanje

Kot vektorsko bazo uporabljamo Azure AI Search, ki je del ekosistema Azure, v katerem že deluje preostali del sistema, in podpira hibridno iskanje brez zunanjih komponent. Iskanje kombinira rezultate polnobesedilnega iskanja BM25 [11] in vektorskega iskanja z algoritmom RRF (angl. *reciprocal rank fusion*) [12]. Kombinacija je za obravnavano domeno pomembna, ker zaposleni pogosto sprašujejo po specifičnih imenih tarifnih paketov, kraticah in kodah storitev, kjer je odločilno natančno besedno ujemanje, vektorska komponenta pa pokrije parafrazirana vprašanja. Semantičnega ponovnega razvrščanja zadetkov ne uporabljamo, ker njegov prispevek pri

trenutnem obsegu podatkov ne odtehta dodatne zakasnitve in stroškov; ostaja kandidat za nadaljnje izboljšave ob bistveni širitvi vsebin.

4.3. Sloj jezikovnih modelov

Agentna aplikacija do jezikovnih modelov ne dostopa neposredno, temveč prek skupnega sloja, ki povezuje dva ponudnika: oblačno storitev Azure AI Foundry, prek katere je na voljo GPT-5, in lokalno gostujoče odprtokodne modele, ki se izvajajo znotraj infrastrukture podjetja. Slednje izpostavljamo prek prehoda Bifrost za umetno inteligenco [13], ki več lokalnih modelov združi pod enotnim vmesnikom, združljivim z vmesnikom OpenAI API, tako da je modele mogoče zamenjati brez posegov v aplikacijo. V času pisanja so bili testirani trije modeli: lokalni model Gemma 4 31B, ki je privzeti, DeepSeek V4 Flash ter GPT-5; uporabnik lahko model izbere za posamezni pogovor.

Uvedba lokalnih modelov ima dva poglavitna razloga. Prvi je zasebnost podatkov: pri lokalno gostujočih modelih vsebina vprašanj in pridobljenih odlomkov ne zapusti infrastrukture podjetja, kar je za sistem, ki bo obdeloval interne podatke, pomembna prednost. Drugi je zmanjšanje stroškov: lokalna inferenca odpravi plačilo po številu žetonov, kar je pri predvideni dnevni uporabi gospodarno. GPT-5 ohranjamo kot najzmogljivejšo možnost za primere, ko razumevanje jezika in sledenje navodilom odtehtata omenjeni prednosti.

4.4. Agentna aplikacija

Agentna aplikacija, zgrajena na ogrodju LlamaIndex, podpira dva načina odgovarjanja, med katerima uporabnik izbira v vmesniku. Oba uporabljata isto orodje za hibridno iskanje in isti jezikovni model, razlikujeta pa se v tem, koliko nadzora nad iskanjem prepustita modelu. Hitri način RAG izvede natanko en klic orodja s poizvedbo, ki je enaka uporabnikovemu vprašanju, in iz pridobljenih odlomkov sintetizira odgovor; ker je zaporedje korakov vnaprej določeno, je hitrejši in primeren za kratka, neposredna vprašanja. Agentni način deluje po vzorcu ReAct [3]: model se sam odloča, ali in kolikokrat bo orodje uporabil ter s kakšno poizvedbo. Pri sestavljenih vprašanjih lahko izvede več zaporednih poizvedb ali neustrezno poizvedbo preoblikuje, pri vprašanjih, ki nadaljujejo prejšnji pogovor, pa odgovori brez iskanja. Privzeti način delovanja je agentni.

Sistemski poziv usmerja agenta k podajanju odgovorov v slovenščini, podpiranju vsakega odgovora z viri in poštenemu priznanju, kadar informacij v bazi znanja ni. Poziv ni vgrajen v izvorno kodo, temveč je upravljan kot različica v orodju Langfuse, kar omogoča iterativno izboljševanje navodil agenta brez ponovne namestitve aplikacije. Sistem podpira več pozivov, prilagojenih vlogi uporabnika: poleg splošnega asistenta je na voljo poziv za tehnike na terenu, ki odgovore oblikuje jedrnato in postopkovno. Agent vire navaja na dva načina: pri krajših odgovorih kot povezave neposredno v besedilu, pri daljših pa kot številčne citate, ki jih vmesnik preslika v ločen razdelek z viri pod odgovorom; obe obliki zagotavljata sledljivost vsake navedbe do izvornega naslova URL.

4.5. Pogovorni vmesnik in podporne storitve

Za pogovorni vmesnik smo izbrali ogrodje Chainlit [14], ki je odprtokodno, neposredno združljivo z ogrodjem LlamaIndex in ponuja izkušnjo v slogu pogovornih asistentov, na katero so uporabniki navajeni iz drugih orodij. Vmesnik omogoča pregled in nadaljevanje preteklih pogovorov, nad vsakim odgovorom pa prikaže korake agenta s klici orodja za iskanje in uporabljenimi poizvedbami, kar razkriva, na podlagi česa je bil odgovor oblikovan. V nastavitvah pogovora uporabnik izbere jezikovni model in sistemski poziv glede na svojo vlogo, nad pogovorom pa preklaplja med načinoma odgovarjanja.

Zgodovina pogovorov, seje in povratne informacije uporabnikov se hranijo v bazi PostgreSQL. Avtentikacija poteka prek storitve Microsoft Entra ID, obstoječega sistema za enotno prijavo v podjetju; skupinsko članstvo uporabljamo za avtorizacijo na ravni aplikacije, dostop pa je omejen na člane namenske skupine. Sistem omogoča nastavljivo beleženje tehničnih sledi izvajanja v orodju Langfuse [15], ki ga je mogoče glede na okolje in namen uporabe vključiti ali izključiti. V fazi razvoja in vrednotenja se beležijo izključno tehnične vsebine, potrebne za

analizo delovanja in kakovosti sistema; sledi ne vsebujejo podatkov naročnikov ali drugih osebnih podatkov ter se uporabljajo zgolj za odkrivanje napak in nadaljnje izboljševanje sistema.

Izbor uporabljenih platform, ogrodij in podpornih komponent je temeljil predvsem na njihovi razširjenosti, podpori, združljivosti s ciljnim okoljem in primernosti za hitro prototipiranje. Z izjemo ogrodij RAG tehnološke alternative niso bile predmet poglobljene primerjalne analize, zato se lahko posamezne komponente pred produkcijsko uvedbo ponovno presodijo in po potrebi zamenjajo.

5 Vrednotenje

5.1. Metodologija

Vrednotenje ni zasnovano kot primerjalna študija algoritmov, temveč kot pragmatična presoja uporabnosti na podlagi realnih vprašanj in izkušenj dejanskih uporabnikov. Opira se na štiri vire podatkov: ročno vrednotenje odgovorov na naboru realnih vprašanj, meritve odzivnih časov iz sledi izvajanja v Langfuse, neposredne povratne informacije uporabnikov in anketo med testnimi uporabniki.

Sistem je pred vrednotenjem približno dva tedna uporabljala skupina sedmih testnih uporabnikov, večinoma iz klicnega centra, njihova vprašanja pa so se beležila v Langfuse. Odgovorov iz tega obdobja za vrednotenje nismo uporabili neposredno, saj pogoji uporabe niso bili nadzorovani: uporabniki so pogosto mešali vsebinsko nepovezana vprašanja v istem pogovoru ali za sestavljena vprašanja izbrali hitri način RAG. Zabeležena vprašanja smo zato ponovno izvedli v nadzorovanih pogojih: vsako v novem pogovoru, brez predhodnega konteksta, v agentnem načinu s privzetim modelom in splošnim sistemskim pozivom. Iz izvoza smo odstranili podvojena in nevsebinska vprašanja, navezujoča se vprašanja preoblikovali v samozadostno obliko, vprašanja brez odgovora v bazi znanja pa namenoma ohranili, saj preverjajo, ali sistem pomanjkanje informacij prizna. Končni nabor obsega 105 vprašanj s področij mobilnih in fiksnih paketov, gostovanja v tujini, postopkov, opreme in storitev.

Vsak odgovor smo uvrstili v eno od petih kategorij: pravilen, delno pravilen, nepravilen, upravičena zavrnitev (sistem sporoči, da nima dovolj informacij, in odgovora v bazi znanja dejansko ni) in neupravičena zavrnitev (sistem odgovor zavrne, čeprav informacija v bazi obstaja). Upravičena zavrnitev je zaželeno vedenje, saj je priznanje omejitve za uporabnika koristnejše od nezanesljivega odgovora. Delež uspešnih odgovorov opredelimo kot delež vprašanj s pravilnim, delno pravilnim ali upravičeno zavrnjenim odgovorom; ob njem poročamo še strožji delež, ki upošteva samo pravilne odgovore. Pravilnost smo presojali glede na dejansko vsebino strani telekom.si v času vrednotenja, ne zgolj glede na vir, ki ga je sistem navedel, s čimer se izognemo krožnosti pri napačno pridobljenih straneh. Odgovore je vrednotil en ocenjevalec, avtor sistema; omejitev delno blažijo vsebinske pripombe uporabnikov, ki so bile pri presoji uporabljene kot dodatna referenca.

5.2. Uspešnost odgovorov

Porazdelitev ocen prikazuje tabela 1. Od 105 vprašanj je sistem na 71 odgovoril pravilno, na 14 delno pravilno, 14 vprašanj je upravičeno zavrnil, 5 neupravičeno, en odgovor pa je bil nepravilen. Delež uspešnih odgovorov znaša 94,3 %, po strožji opredelitvi pa 67,6 %.

Struktura neuspešnih odgovorov je povednejša od samih deležev. Očitno napačen odgovor, torej tak, ki bi uporabnika zavedel z napačno informacijo, je bil en sam: na vprašanje o cenovnem območju gostovanja za Karibe je sistem samozavestno in z viri opisal zakupe prenosa podatkov, torej odgovoril na drugo vprašanje. Primer ponazarja najnevarnejšo obliko napake: odgovor je tekoč, strukturiran in podprt z viri, zato deluje verodostojno. Delno pravilni odgovori so praviloma nepopolni, ne napačni, in manjkajoče informacije pošteno označijo. Neupravičene zavrnitve nastanejo, kadar korak iskanja ne pridobi relevantne strani, praviloma pri kratkih poizvedbah z imeni ponudb, ki se prekrivajo z drugimi vsebinami; agent je pri tem ravnal skladno z navodili, saj si odgovora ni izmislil, temveč je ponudil sorodne zadetke in prosil za natančnejši opis. Upravičene zavrnitve

večinoma izvirajo iz vprašanj, ki zahtevajo osebne podatke strank ali interne vsebine, torej informacije, ki jih trenutna baza znanja načrtno ne vsebuje; sistem je omejitev priznal in uporabnika usmeril na ustrezne portale.

Tabela 1: Rezultati ročnega vrednotenja odgovorov na naboru 105 vprašanj.

Ocena	Število vprašanj	Delež [%]
pravilen	**71**	**67,6**
delno pravilen	**14**	**13,3**
nepravilen	**1**	**1,0**
upravičena zavrnitev	**14**	**13,3**
neupravičena zavrnitev	**5**	**4,8**
delež uspešnih odgovorov	**99**	**94,3**

5.3. Odzivni časi

Odzivne čase smo izmerili iz sledi izvajanja v Langfuse za vseh 105 vprašanj; rezultate povzema tabela 2. Poročamo mediano in razpon med 5. in 95. percentilom, saj je porazdelitev desno asimetrična. Ker se odgovor uporabniku prikazuje pretočno, merimo tudi čas do prvega žetona odgovora (angl. *time to first token*) kot mero zaznane odzivnosti.

Tabela 2: Odzivni časi po korakih izvajanja agenta za 105 vprašanj iz nabora za vrednotenje. Meji razpona sta 5. in 95. percentil.

Korak	Mediana [s]	Razpon [s]
sklepalni koraki (odločitev in poizvedba)	**0,6**	**0,5–0,9**
iskanje po vektorski bazi	**0,9**	**0,7–1,2**
končno oblikovanje odgovora	**11,7**	**4,6–21,4**
skupaj	**13,8**	**6,0–23,0**
čas do prvega žetona odgovora	**5,3**	**4,3–6,2**

Skupni čas skoraj v celoti določa generiranje besedila v jezikovnem modelu; odločanje agenta in iskanje po vektorski bazi skupaj tipično zahtevata približno 1,5 sekunde. Agentna zasnova in poizvedovanje po bazi znanja k skupnemu odzivnemu času prispevata razmeroma majhen delež, zato bi se morebitne optimizacije morale osredotočiti na generiranje, na primer z izbiro hitrejšega modela ali s krajšimi odgovori. Agent je pri 97 vprašanjih orodje za iskanje klical enkrat, pri treh dvakrat, pri petih pa je odgovoril brez iskanja, praviloma tedaj, ko je vprašanje zahtevalo podatke, ki jih baza znanja načrtno ne vsebuje; to potrjuje, da število iskanj prilagaja vprašanju. Z vidika uporabnika je pomembnejši čas do prvega žetona: uporabnik začne odgovor brati po približno petih sekundah, celoten odgovor pa je tipično izpisan v slabih štirinajstih sekundah. Rezultati nakazujejo, da je taka odzivnost sprejemljiva tudi za delo v klicnem centru; skladno s tem je pet od sedmih anketirancev poročalo o krajšem ocenjenem času iskanja informacij kot pred uvedbo sistema, nihče pa o daljšem.

5.4. Povratne informacije uporabnikov

Splošno oceno uporabnosti smo zajeli z anonimno anketo med testnimi uporabniki, ki jo je do zaključka pisanja izpolnilo sedem uporabnikov. Zaradi majhnega vzorca rezultatov ne posplošujemo; poročamo porazdelitve in mediane. Anketiranci sistem uporabljajo za preverjanje informacij o paketih, zakupih, cenikih, postopkih in pogojih, torej natanko za naloge, ki jih je zajem zahtev predvidel. Pred uvedbo sistema so odgovore iskali po več virih hkrati (spletna stran, interni portal, ceniki, elektronska pošta, sodelavci); čas iskanja odgovora je šest anketirancev ocenilo na 1 do 5 minut, eden na 5 do 15 minut, s sistemom pa pet anketirancev odgovor najde v manj kot minuti, dva pa v 1 do 5 minutah. Pri petih anketirancih se je ocenjeni čas iskanja torej skrajšal, pri nobenem pa podaljšal.

Tabela 3: Porazdelitev odgovorov na trditve ankete
(1 pomeni sploh se ne strinjam, 5 popolnoma se strinjam), N = 7.

Trditev	1	2	3	4	5	Mediana
Sistem mi je pogosto vrnil uporaben odgovor.	**0**	**0**	**3**	**4**	**0**	**4,0**
Navedeni viri so podpirali odgovor.	**0**	**1**	**1**	**4**	**1**	**4,0**
Odgovorom sistema zaupam.	**1**	**2**	**3**	**1**	**0**	**3,0**
Sistem mi prihrani čas pri iskanju informacij.	**0**	**0**	**2**	**4**	**1**	**4,0**
Sistem bi priporočil sodelavcem.	**0**	**1**	**1**	**4**	**1**	**4,0**

Kot prikazuje tabela 3, so uporabnost odgovorov, ustreznost virov, prihranek časa in pripravljenost priporočiti sistem ocenjeni zmerno pozitivno z mediano 4,0, najnižje pa je ocenjeno zaupanje v odgovore, z mediano 3,0 in s šestimi od sedmih ocen na sredini lestvice ali pod njo. Ta razkorak je vsebinsko pomenljiv in skladen s strukturo napak iz tabele 1: čeprav so vsebinske napake redke, so uporabniki med testiranjem naleteli nanje in na vsebinske vrzeli, vsaka taka izkušnja pa zahteva ponovno preverjanje tudi sicer pravilnih odgovorov. Odgovori na odprta vprašanja razkorak pojasnijo: kot prednosti uporabniki navajajo hitrost, preglednost odgovorov in združevanje informacij, kot pomanjkljivosti pa zahtevnejša sestavljena vprašanja, nepopolne sezname možnosti, zastarele podatke in vsebinske vrzeli. Med želenimi izboljšavami prevladujejo interne vsebine, ki jih trenutna baza znanja ne pokriva. Takšno stališče povzema odgovor ene od anketirank: „AI chat kot tak je vrhunska rešitev, ker prideš zelo hitro do odgovora, dodelati je treba samo bazo, iz katere črpa informacije.“

6 Zaključek

V prispevku smo predstavili zasnovo, izvedbo in vrednotenje agentnega sistema RAG za podporo zaposlenim v Telekomu Slovenije. Sistem sestavljata cevovod za zajem podatkov, ki vsebine javne spletne strani pretvarja v vektorske vložitve v bazi Azure AI Search, in agentna aplikacija na ogrodju LlamaIndex, ki po vzorcu ReAct sama odloča o hibridnem iskanju in vsak odgovor podpre z viri. Poleg oblačnega modela sistem prek prehoda Bifrost uporablja lokalno gostujoče jezikovne modele; pri njihovi uporabi vsebina vprašanj in pridobljenih odlomkov ne zapusti infrastrukture podjetja, na kateri tečejo tudi vse lastne komponente sistema.

Vrednotenje na 105 realnih vprašanjih testnih uporabnikov je po vnaprej določeni širši opredelitvi uspešnosti pokazalo 94,3 % uspešnih odgovorov, po strožji opredelitvi, ki upošteva samo v celoti pravilne odgovore, pa 67,6 %. Očitno napačen odgovor je bil en sam. Čas do prvega žetona je znašal približno pet sekund, ocena uporabnikov pa je bila zmerno pozitivna, pri čemer je bil glavni zadržek zaupanje v odgovore. Izkušnje potrjujejo tri praktične ugotovitve, prenosljive na sorodne sisteme: da agentno odločanje in iskanje k odzivnemu času prispevata razmeroma majhen delež v primerjavi z generiranjem, da je za sprejem sistema pomembnejša pokritost baze znanja

kot izpopolnjevanje samega odgovarjanja ter da uporabniki brez uvajanja funkcionalnosti, kot je izbira načina odgovarjanja, ne uporabljajo optimalno. Smeri nadaljnjega dela izhajajo neposredno iz teh ugotovitev: vključitev internih virov, predvsem Telekomove interne baze znanja, skupaj z nadzorom dostopa po uporabniških skupinah, dogodkovno proženje zajema prek spletnega sprožilca sistema CMS, sistematična primerjava razpoložljivih jezikovnih modelov in samodejno vrednotenje na pripravljenem naboru vprašanj ter vključitev v okolje Microsoft Teams.

Literatura

[1] BROWN Tom B. in drugi, "Language Models are Few-Shot Learners", Advances in Neural Information Processing Systems, 2020.

[2] LEWIS Patrick in drugi, "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks", Advances in Neural Information Processing Systems, 2020.

[3] YAO Shunyu, ZHAO Jeffrey, YU Dian, DU Nan, SHAFRAN Izhak, NARASIMHAN Karthik, CAO Yuan, "ReAct: Synergizing Reasoning and Acting in Language Models", International Conference on Learning Representations (ICLR), 2023.

[4] ASAI Akari, WU Zeqiu, WANG Yizhong, SIL Avirup, HAJISHIRZI Hannaneh, "Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection", International Conference on Learning Representations (ICLR), 2024.

[5] YAN Shi-Qi, GU Jia-Chen, ZHU Yun, LING Zhen-Hua, "Corrective Retrieval Augmented Generation", arXiv:2401.15884, 2024.

[6] LangChain Documentation, https://docs.langchain.com/, obiskano 12. 7. 2026.

[7] LlamaIndex Documentation, https://developers.llamaindex.ai/python/framework/, obiskano 12. 7. 2026.

[8] Haystack Documentation, https://docs.haystack.deepset.ai/, obiskano 12. 7. 2026.

[9] Semantic Kernel Documentation, https://learn.microsoft.com/en-us/semantic-kernel/, obiskano 12. 7. 2026.

[10] Crawl4AI: Open-Source LLM-Friendly Web Crawler and Scraper, https://github.com/unclecode/crawl4ai, obiskano 12. 7. 2026.

[11] ROBERTSON Stephen, ZARAGOZA Hugo, "The Probabilistic Relevance Framework: BM25 and Beyond", Foundations and Trends in Information Retrieval, letnik 3, številka 4, 2009, str. 333–389.

[12] CORMACK Gordon V., CLARKE Charles L. A., BÜTTCHER Stefan, "Reciprocal Rank Fusion Outperforms Condorcet and Individual Rank Learning Methods", Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2009.

[13] Bifrost: Open-Source LLM Gateway, https://github.com/maximhq/bifrost, obiskano 12. 7. 2026.

[14] Chainlit: Build Conversational AI Applications, https://docs.chainlit.io/, obiskano 12. 7. 2026.

[15] Langfuse: Open-Source LLM Engineering Platform, https://langfuse.com/docs, obiskano 12. 7. 2026.