Dr. sc. Darko Andročec[1]

IZRADA PROTOTIPA WEB APLIKACIJE KORIŠTENJEM AGENTSKOG AI ALATA

Stručni rad / Professional paper
UDK / UDC: 004.42:004.8
DOI: 10.51650/ezrvs.20.1-2.1
Primljeno / Received: 23/04/2026
Prihvaćeno / Accepted: 12/05/2026

U ovom radu istražene su mogućnosti izrade prototipa web-aplikacija korištenjem agentskog AI alata Claude Code te su uspoređene s postojećim pristupom malokodnog programiranja u Oracle Apex okruženju koje se koristi na laboratorijskim vježbama iz predmeta Projektiranje informacijskih sustava na Fakultetu organizacije i informatike. Kao slučaj korištenja odabran je ERA model R1 računa sa šest entiteta, pri čemu je alatu kao ulazni podatak proslijeđena samo slikovna datoteka ERA modela. Claude Code je na temelju toga samostalno generirao relacijsku bazu podataka, popunio ju testnim podacima te izradio funkcionalnu web aplikaciju u Python/Flask/SQLite tehnološkom okviru s CRUD formama za sve entitete, izvještajima u PDF obliku, grafikonima i REST servisima koji vraćaju podatke u JSON formatu. Rad prikazuje korištene upite, odabrane tehnologije, dodatne funkcionalnosti implementirane u iterativnim koracima te eksperiment prebacivanja aplikacije u Oracle Apex okruženje. Rezultati pokazuju da je prototip osnovne aplikacije izrađen približno četiri puta brže (procjena temeljena na praktičnom iskustvu autora i vremenu mjerenom u promatranom slučaju jednostavnijeg ERA modela sa šest entiteta) nego ručnom izradom u Oracle Apexu, uz mogućnost jednostavnog proširenja funkcionalnostima koje u Apexu nije trivijalno realizirati, dok se pri prebacivanju u Oracle Apex pojavljuju problemi prilikom uvoza generirane aplikacijske SQL skripte.

Ključne riječi: *agentski AI; Claude Code; prototip web aplikacije; vibracijsko kodiranje; Oracle Apex.*

1. Uvod

Cilj ovog rada jest istražiti mogućnosti izrade prototipa web aplikacija korištenjem agentskog AI alata. Slučaj korištenja jest uporaba alata *Claude Code* za izradu prototipa web aplikacije za R1 račun. Taj se primjer već koristi u nastavi na predmetu Projektiranje informacijskih sustava koji je obavezan predmet na drugoj godini prijediplomskog studija na Fakultetu organizacije i informatike. Trenutno se za izradu tog prototipa u sklopu laboratorijskih vježbi

[1] Izvanredni profesor; Sveučilište u Zagrebu, Fakultet organizacije i informatike, Pavlinska 2, 42000 Varaždin, Republika Hrvatska; e-mail: dandrocec@foi.unizg.hr

koristi *Oracle Apex*, sustav u oblaku za malokodno kodiranje temeljeno na bazi podataka. Cilj je utvrditi prednosti i nedostatke korištenja alata *Claude Code* u odnosu na postojeće korištenje alata *Oracle Apex*. Autor pokušava implementirati sve mogućnosti koje se inače rade u tom dijelu laboratorijskih vježbi. Riječ je o jednostavnijem ERA modelu o R1 računu s ukupno šest entiteta.

Istraživačka pitanja su sljedeća: može li *Claude Code* napraviti funkcionalnu CRUD (engl. *Create*, *Read*, *Update*, *Delete*; kreiranje, čitanje, ažuriranje i brisanje zapisa) web aplikaciju na temelju slike ERA modela? Pri tome ta aplikacija treba imati CRUD forme za sve tablice iz baze podataka, nekoliko izvještaja i grafikona te nekoliko *RESTful (*engl. *Representational State Transfer;* programska sučelja temeljena na HTTP protokolu) servisa koji vraćaju podatke iz tablica u JSON (engl. *JavaScript Object Notation*; tekstualni format za razmjenu strukturiranih podataka) formatu. Može li se korištenjem agentskog AI alata traženi prototip napraviti brže nego korištenjem alata za malokodno programiranje poput *Oracle Apexa*? Koje dodatne funkcionalnosti autor može u prototipu implementirati, a da ih nije mogao jednostavno u *Oracle Apexu*?

Metodologija rada temelji se na jednom slučaju korištenja s ciljem ilustracije pristupa. Kao ulazni podatak alatu *Claude Code* dostavljena je samo slikovna .png datoteka ERA modela R1 računa, dok je popis traženih funkcionalnosti zadan tekstualnim upitima dostupnima u poglavlju 3. Korišten je *Claude Code* (verzija konzolnog klijenta iz travnja 2026. godine, podrazumijevani jezični model *Sonnet 4.6*) uz Pro pretplatu na uslugu *Claude AI*; usporedna implementacija u alatu *Oracle Apex* napravljena je u verziji 24.2.14 okruženja apex.oracle.com. Vrijeme izrade mjereno je ručno, štopericom, od trenutka slanja pojedinog upita do trenutka kada je alat prijavio dovršenje zadatka, čemu je pridodano i vrijeme ručnog testiranja u pregledniku. Kao kriteriji uspješnosti korišteni su: (a) uspješno generiranje baze podataka i CRUD formi za svih šest entiteta iz ERA modela R1 računa; (b) uspješno generiranje barem tri PDF izvještaja, dva grafikona i nekoliko REST servisa u JSON formatu; (c) prolaz svih ručno provedenih CRUD scenarija nad svakom tablicom; (d) prolaz svih automatski generiranih *pytest* testova. Za svaku osnovnu funkcionalnost provedena je jedna iteracija s alatom, a u slučajevima kada je nakon ručnog testiranja uočena pogreška provedeno je do tri dodatnih iteracija ispravljanja. Razina ljudske intervencije autora ograničena je na pisanje upita, odobravanje pojedinih koraka koje alat traži (npr. potvrdu kreiranja datoteka i pokretanja naredbi), pokretanje aplikacije te ručno klikanje kroz korisničko sučelje radi provjere funkcionalnosti, dok je sav programski kod, struktura baze podataka, datoteke konfiguracije i tekstovi predložaka generirani autonomno radom alata *Claude Code*.

2. Pregled literature

Wang i sur. (2025) sistematiziraju područje agentskog programiranja, gdje veliki jezični modeli (*LLM*-ovi) autonomno planiraju, izvršavaju i iteriraju softverske zadatke uz minimalnu ljudsku intervenciju. Autori klasificiraju postojeće tehnike prema arhitekturi agenata, mehanizmima korištenja alata i strategijama interakcije s razvojnim okruženjem. Identificiraju ključne izazove poput pouzdanosti, evaluacije i održavanja dugoročnog konteksta. Rad služi kao temeljna referenca za razumijevanje trenutnog stanja i otvorenih istraživačkih pitanja u

agentskom kodiranju. Plaat i sur. (2025) daju širi pregled agentskih *LLM*-ova izvan konteksta kodiranja, obuhvaćajući rasuđivanje, djelovanje i interakciju s okolinom. Posebna pažnja posvećena je problemima evaluacije i sigurnosti autonomnih sustava. Wei i sur. (2026) usredotočuju se na mehanizme rasuđivanja koji omogućuju agentskim *LLM*-ovima rješavanje složenih zadataka s mnogo koraka. Autori predstavljaju tehnike koje kombiniraju lančano rasuđivanje s dinamičkim pozivanjem vanjskih alata i pretraživanja.

Ge i sur. (2025) uvode i formaliziraju pojam vibracijsko kodiranje (engl. *vibe coding*) — paradigmu u kojoj razvojni programeri surađuju s *LLM*-ovima kroz intuitivne upute visoke razine umjesto preciznog specificiranja. Autori analiziraju alate, radne tokove i kognitivne promjene koje ta paradigma donosi u praksu softverskog inženjerstva. Raspravljaju se i rizici poput gubitka kontrole nad kvalitetom koda i održivosti. Pan i sur. (2024) predstavljaju *SWE-Gym*, okruženje za treniranje agenata softverskog inženjerstva na realnim problemima iz *GitHub* repozitorija. Vergopoulos i sur. (2025) bave se problemom zasićenja i curenja podataka kod postojećih alata za mjerenje performansi poput *SWE-Bench*. Autori predlažu metodiku automatskog generiranja novih zadataka na razini repozitorija, čime se osigurava kontinuirano i nepristrano ocjenjivanje agenata.

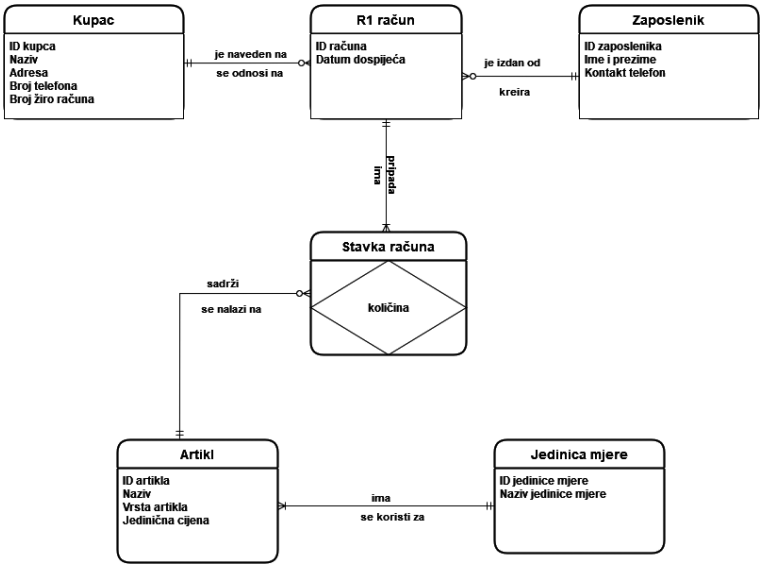
Tian i sur. (2026) proširuju *SWE-Bench* paradigmu na domenu mobilnih aplikacija, testirajući mogu li agenti razvijati industrijski primjenjiv *Android* i *iOS* kod. Rad identificira mobilni razvoj kao važnu, a slabo pokrivenu domenu za AI agentske sustave. Xu i sur. (2025) predlažu *SWE-Compass* kao objedinjeni okvir za evaluaciju agentskih kodirajućih sposobnosti, koji pokriva različite jezike, domene i razine složenosti zadataka. Autori ističu nedostatke postojećih sustava za mjerenje performansi koji se fokusiraju na uske scenarije, najčešće *Python* i popravljanje grešaka u programskom kodu. Kroz višedimenzionalno ocjenjivanje omogućuje se dublje razumijevanje stvarnih prednosti i slabosti pojedinih modela i agenata. Rad predstavlja korak prema reprezentativnijem mjerenju napretka u agentskom kodiranju.

Navedeni radovi zajedno ocrtavaju brzo sazrijevanje područja agentskog programiranja — od konceptualnih pregleda (Wang i sur., 2025; Plaat i sur., 2025; Wei i sur., 2026) preko novih razvojnih paradigmi (Ge i sur., 2025) do konkretnih metoda treniranja (Pan i sur., 2024) i sve sofisticiranijih sustava evaluacije (Vergopoulos i sur., 2025; Tian i sur., 2026; Xu i sur., 2025). Glavni trend jest pomak od statičkih modela prema autonomnim agentima sposobnima za rasuđivanje u više koraka i rad u realnim softverskim okruženjima, uz istovremeni rast potrebe za robusnim i raznovrsnim mjerenjima performansi. Literatura zasad ne podupire tvrdnju da su današnji agenti za programiranje zreli kao opća zamjena za programera, ali jasno pokazuje da su postali ozbiljan istraživački i inženjerski alat. Najvažniji otvoreni problem nije samo sposobnost generiranja koda, nego pouzdanost pri dugotrajnim, višekratnim i stvarnim zadacima (Wang i sur., 2025; Plaat i sur., 2025; Wei i sur., 2026; Ge i sur., 2025; Pan i sur., 2024; Vergopoulos i sur., 2025; Tian i sur., 2026).

3. Slučaj korištenja

Prvi slučaj korištenja jest primjer R1 računa koji se koristi na laboratorijskim vježbama iz predmeta Projektiranje informacijskih sustava, te se taj ERA model implementira u okruženju *Oracle Apex*. To se radi kroz četiri termina laboratorijskih vježbi od kojih svaki traje 45 minuta.

Slika 1. ERA model R1 računa



Izvor: izrada autora

Ovaj ERA model prikazuje sustav izdavanja R1 računa kroz pet jakih entiteta i jedan slabi entitet koji su međusobno povezani, gdje središnju ulogu ima R1 račun koji služi kao poveznica između sudionika i samog predmeta prodaje. Proces započinje relacijom između kupca i računa, pri čemu jedan kupac može imati više računa, dok se svaki račun odnosi na točno jednog kupca s definiranim ID-om, nazivom i podacima o žiro računu. Paralelno s tim, zaposlenik (identificiran imenom i kontaktom) kreira račun, što čini vezu u kojoj jedan djelatnik može izdati mnoštvo računa. Sam račun se dalje raščlanjuje na stavke računa, koje funkcioniraju kao slabi entitet i sadrže ključni podatak o količini; svaka stavka pripada jednom računu, ali jedan račun redovito sadrži više stavki. Te stavke izravno su povezane s entitetom artikl, koji nosi podatke o nazivu i jediničnoj cijeni, a svaki je artikl dodatno definiran pripadajućom jedinicom mjere (poput komada ili kilograma) kako bi se osigurala preciznost u iskazivanju prodane robe ili usluga.

3.1. Kreiranje baze podataka i CRUD formi

Prvi korak jest kreiranje baze podataka, popunjavanje tablica početnim testnim podacima te kreiranje CRUD formi u aplikaciji koje pokrivaju svih šest opisanih entiteta. Preduvjet je, naravno, instaliran *Claude Code*, pri čemu je prilikom izrade ovog rada korištena konzolna aplikacija. Da bi *Claude Code* radio, treba ga povezati s korisničkim računom *Claude AI*. Korištena je *Pro* verzija računa koja ima određena ograničenja po broju dopuštenih *tokena*, te je povremeno bilo potrebno stati i nastaviti kasnije kad bi se dosegnulo ograničenje (korisnički račun *Claude AI Pro* ima ograničenje po terminu od pet sati te kumulativno tjedno ograničenje). Tijekom rada na ovom radu kumulativno tjedno ograničenje nije bilo prekoračeno, ali u nekoliko slučajeva došlo je do probijanja granice dozvoljene u svakom intervalu od pet sati, pa je povremeno trebalo pričekati. Sve korištene upite (engl. *prompt*) postavljene alatu *Claude Code* autor u cijelosti stavlja u

rad, a sav generirani kod dostupan je na javno otvorenom *GitHub* repozitoriju.[2] Kako bi se krenulo s generiranjem baze podataka i web aplikacije, kreiran je novi prazan direktorij u koji je stavljena samo slikovna .png datoteka ERA modela R1 računa prikazanog u ovom radu na slici 1. U komandnom načinu rada otvara se *Claude Code* u tom direktoriju, te se pišu tekstualni upiti AI alatu. Za početak se koristi podrazumijevani model *Sonnet 4.6*. Inače se može koristiti i moćniji model ukoliko bi bila riječ o složenijem zadatku (npr. puno većem ERA modelu), ali onda taj AI model puno brže koristi dostupnu kvotu u sklopu *Claude Pro* pretplate. Prvi upit glasi:

> *Napravi bazu podataka i popuni tablice s testnim podacima. Nakon toga napravi web aplikaciju s CRUD formama za cijelu ovu bazu podataka. Aplikacija neka bude na hrvatskom jeziku. U datoteku tehnologije.txt napiši koje si tehnologije koristio za kreiranje opisanog slučaja korištenja.*

Claude Code kreće s radom i povremeno postavlja pitanja za odobrenje pojedinih koraka, npr. potvrđivanje kreiranja baze podataka, početne stranice web aplikacije i slično. U samom je upitu alatu ostavljen izbor tehnologije za implementaciju ovog slučaja korištenja. *Claude Code* je na kraju izabrao programski jezik *Python*, web okvir *Flask* te bazu podataka *SQLite*, što je jedan od najčešćih izbora za ovako jednostavnu web aplikaciju. Detaljan opis korištenih tehnologija prikazan je u tablici 1.

Tablica 1. Opis odabranih tehnologija za implementaciju web aplikacije o R1 računima

Br.	Kategorija	Tehnologija / Alat	Opis
1.	Programski jezik	*Python* 3.14.3	*Python* je glavni programski jezik korišten za logiku na strani poslužitelja i pristup bazi podataka.
2.	Web okvir	*Flask* 3.1.3	Jednostavan web okvir za *Python*.
3.	Baza podataka	*SQLite* 3	Ugrađena relacijska baza podataka kojoj se pristupa putem *Python sqlite3* modula (standardna biblioteka).
4.	Frontend tehnologije	*HTML5*, *Bootstrap* 5.3.0, *Bootstrap Icons* 1.11.0, *CSS3*, *JavaScript* (ES6+)	*HTML5* – strukturiranje stranica. *Bootstrap* 5.3 – responzivni dizajn. *Bootstrap Icons* – ikone korisničkog sučelja. *CSS3* – vlastiti stilovi navigacije i kartica. *JavaScript* – dinamički izračun iznosa stavke i potvrda brisanja.
5.	Arhitektura aplikacije	MVC obrazac, CRUD operacije	Model – *SQLite* baza podataka i init_db.py. View – *Jinja2* HTML predlošci (templates/). Controller – *Flask* rute u app.py. CRUD (Create, Read, Update, Delete) implementiran za svih šest entiteta baze podataka.

Izvor: izrada autora

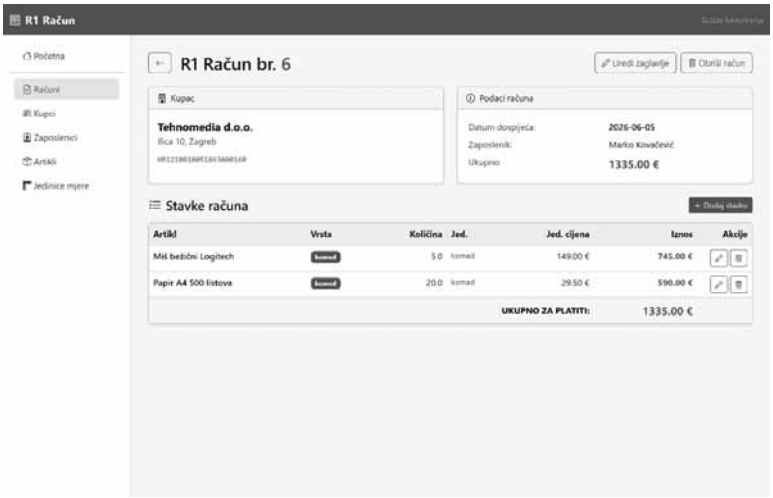
Nakon toga autor je ručno provjerio kreiranu web aplikaciju koja je pokrenuta na adresi http://localhost:5000/, te je utvrđeno da sve CRUD forme rade (kreiranje, mijenjanje, brisanje i dodavanje podataka u svaku tablicu iz početne slike ERA modela), te da svaka tablica ima

[2] https://github.com/dandrocec/r1_racun_claude_code

popunjene testne podatke. Slika 2 prikazuje detalje jednog R1 računa s dvije pripadajuće stavke računa. Izrada ovog dijela aplikacije u alatu *Claude Code* s ručnim testiranjem trajala je oko 30 minuta.

Konkretno su, klikanjem kroz korisničko sučelje aplikacije pokrenute na adresi *http://localhost:5000/,* provjerene sljedeće CRUD operacije nad svakom od šest tablica (*Kupac, Zaposlenik, Artikl, Jedinica_mjere, Račun i Stavka_računa*): unos novog zapisa kroz formu, prikaz liste svih zapisa, otvaranje detaljnog prikaza pojedinog zapisa, izmjena postojećeg zapisa i brisanje zapisa, što odgovara ukupno 30 ručno provjerenih CRUD scenarija. Tijekom ručnog testiranja u promatranom slučaju nije zabilježena nijedna funkcionalna pogreška u izvedbi CRUD operacija, a integritet referencijalnih ograničenja (npr. nemogućnost brisanja kupca koji ima račun) potvrđen je i ručno i kasnije generiranim *pytest* testovima.

Slika 2. Prikaz jednog R1 računa u generiranoj aplikaciji



Izvor: izrada autora

3.2. Izvještaji, grafikoni i REST servisi

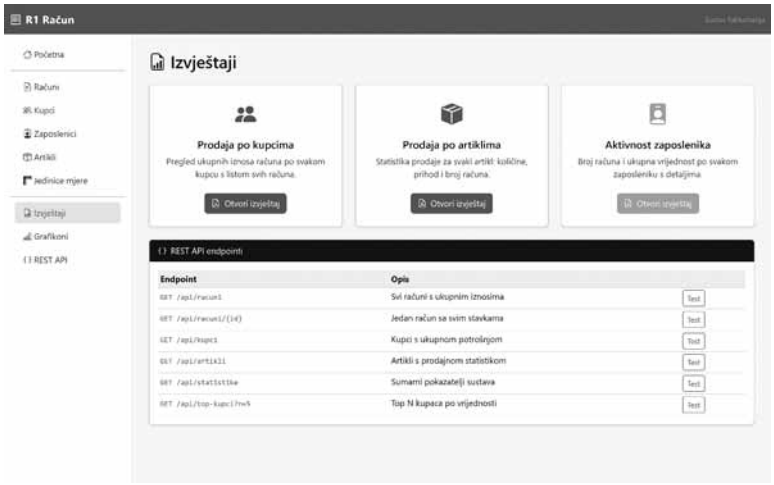
U sklopu laboratorijskih vježbi u alatu *Oracle Apex*, na ovom primjeru ERA modela još se rade izvještaji koji koriste poglede s atributima iz više tablica iz baze podataka, uz korištenje ugrađenih čarobnjaka i formi za grafikone definiranih u samom alatu. Također, na sličan se način kreira i nekoliko grafikona te REST servisa za prikaz podataka iz tablica. Slično se sada pokušava u alatu *Claude Code* sljedećim upitom:

> *Dodaj barem tri izvještaja koji ispisuju podatke iz više tablica iz baze podataka. Izvještaj neka se prikazuje u PDF obliku (engl. Portable Document Format; format prijenosnog dokumenta) i neka bude spreman za ispis. Osim toga, dodaj barem dva grafikona. Dodaj i nekoliko smislenih REST servisa kojima se podaci iz baze prikazuju u JSON formatu.*

Ovaj je dio *Claude Code* napravio brzo, otprilike za 10 minuta, plus dodatnih 10–15 minuta za ručno testiranje. Vremenski je za ovaj osnovni dio aplikacije, koji se radi na laboratorijskim vježbama na predmetu Projektiranje informacijskih sustava, potrebno približno četiri puta manje vremena (procjena temeljena na praktičnom iskustvu autora i izmjerenom vremenu izrade u promatranom slučaju) nego u alatu *Oracle Apex*. Pogotovo prvi dio, tj. kreiranje baze podataka iz slikovne datoteke ERA modela, što se u alatu *Oracle Apex* treba raditi ručno. Dvije glavne web stranice o izvještajima, REST servisima i grafikonima prikazane su na slikama 3. i 4.

Konkretno su izrađena tri PDF izvještaja: (1) izvještaj liste svih računa s pripadajućim stavkama i ukupnim iznosima, (2) izvještaj prometa po kupcima i (3) izvještaj prodanih artikala s količinama. Svaki je izvještaj otvoren u pregledniku te je provjereno da se ispravno prikazuje i može poslati na ispis. Dodana su i dva grafikona prikazana pomoću *JavaScript* biblioteke *Chart.js*: stupčasti grafikon prometa po mjesecima i tortni grafikon udjela pojedinih artikala u prometu. Izrađeno je pet REST krajnjih točaka koje vraćaju podatke u JSON formatu: /api/kupci, /api/racuni, /api/artikli, /api/stavke i /api/zaposlenici, koje su provjerene i alatom *curl* i u pregledniku. Tijekom ručnog testiranja izvještaja, grafikona i REST servisa u promatranom slučaju nisu uočene funkcionalne pogreške; jedina dodatna prilagodba bila je ispravak formata datuma u jednom od izvještaja, koja je riješena u jednoj dodatnoj iteraciji s alatom *Claude Code*.

Slika 3. Generirana web stranica s izvještajima i REST servisima



Izvor: izrada autora

Slika 4. Grafikoni napravljeni nakon upita u alatu Claude Code



Izvor: izrada autora

3.3. Dodatne funkcionalnosti

Nakon toga moguće je implementirati različite dodatne funkcionalnosti. Kada osnovni radni prototip funkcionira, mogu se raditi nove iteracije, jedna po jedna. Svaki od primjera upita i iteracija u nastavku traje 2–5 minuta da se implementira u alatu *Claude Code*:

- *Dodaj sustav prijave. Samo prijavljeni zaposlenici mogu kreirati račune. Administrator može upravljati korisnicima.*
- *Dodaj gumb za izvoz liste računa u Excel (npr. openpyxl). Dodaj i filtriranje po datumu.*
- *Dodaj mogućnost storniranja računa (ne briše se, nego se označava kao storniran s datumom i razlogom).*
- *Dodaj slanje računa na e-mail kupca kao PDF prilog koristeći Flask-Mail.*
- *Kreiraj Dockerfile i docker-compose.yml za kontejnerizaciju.*

Dobra je opcija napraviti i automatsko testiranje, npr. korištenjem sljedećeg upita u alatu *Claude Code*:

Napravi pytest testove koji pokrivaju: kreiranje svakog entiteta, provjeru FK relacija, izračun ukupnog iznosa računa, generiranje broja računa, CRUD operacije za svaki entitet, dodavanje stavke na račun i provjeru kaskadne zaštite (da se ne može obrisati kupac koji ima račun). Koristi in-memory SQLite i seed podatke u conftest.py. Pokreni testove i rezultate zapiši u datoteku test_results.txt.

3.4. Prebacivanje generirane aplikacije u Oracle Apex

Sljedeći je eksperiment prebacivanje generirane aplikacije u *Oracle Apex*. Odnosno, zadavanje alatu *Claude Code* da napravi web aplikaciju sa sličnim mogućnostima i nad istom bazom podataka, kako bi se vidjelo može li to napraviti. Svi su ovi upiti napravljeni u travnju 2026. godine. Modeli i sam alat *Claude Code* stalno napreduju, pa je moguće da će u bliskoj budućnosti biti riješen problem koji je sada identificiran. U alat *Claude Code* stavljen je sljedeći upit:

Napravi web aplikaciju sa sličnim funkcionalnostima i istom bazom podataka i testnim podacima za Oracle Apex. Koristim cloud verziju Oracle Apexa (apex.oracle.com) verzije 24.2.14.

Nakon 31 minute rada potrošena je sva petosatna kvota za *Claude Code* u sklopu *Pro* pretplate. Nastavljeno je nakon isteka tog vremena, te je *Claude Code* završio za dodatne četiri minute. Pripremio je sve elemente i upute za redoslijed instalacije. U *SQL Workshopu* alata *Oracle Apex* učitane su dvije SQL skripte, jedna za kreiranje baze podataka, a druga za popunjavanje tablica testnim podacima. Taj je dio prošao bez problema. Također je bez problema prošao i uvoz SQL skripte za REST servise nad tablicama baze podataka.

Međutim, problemi odnosno greške javili su se prilikom pokušaja uvoza SQL skripte za samu aplikaciju. Alatu *Claude Code* prosljeđivani su upiti da se prilikom uvoza dogodila greška te su kopirani naziv greške i cjelokupan opis koji je vratio *Oracle Apex* sustav. Unatoč višekratnim iteracijama, problem nije u potpunosti otklonjen.

4. Zaključak

Provedeno istraživanje potvrđuje da agentski AI alat *Claude Code* može samostalno, na temelju slikovne datoteke ERA modela, generirati funkcionalan prototip web aplikacije koji zadovoljava sve zahtjeve postavljene kao istraživačka pitanja — CRUD forme za sve tablice, izvještaje u PDF obliku, grafikone te *RESTful* servise u JSON formatu. Najveća prednost ovakvog pristupa u odnosu na *Oracle Apex* očituje se u vremenu izrade, koje je za osnovni dio prototipa smanjeno približno četiri puta (u promatranom slučaju jednostavnijeg ERA modela R1 računa sa šest entiteta i u kontekstu izrade nastavnog prototipa, te uz napomenu da je riječ o procjeni temeljenoj na praktičnom iskustvu autora), prvenstveno zato što kreiranje baze podataka iz ERA modela u alatu *Oracle Apex* treba raditi ručno, dok ga agentski alat obavi automatski iz slikovne datoteke. Dodatna prednost jest i šira paleta funkcionalnosti koje se mogu implementirati kratkim iterativnim upitima — od autentifikacije i autorizacije, preko izvoza u *Excel*, storniranja računa i slanja PDF-a elektroničkom poštom, pa sve do kontejnerizacije aplikacije i automatskog testiranja korištenjem alata *pytest* — a koje u *Oracle Apex* okruženju nisu dostupne odnosno zahtijevaju znatno veći napor implementacije. Iznesene zaključke valja tumačiti u promatranom slučaju jednostavnijeg ERA modela i u kontekstu izrade spomenutog prototipa, dok njihovu generalizaciju na složenije poslovne sustave i drugačije razvojne kontekste tek treba dodatno ispitati.

S druge strane, uočena su i određena ograničenja. Korištenje *Pro* pretplate za *Claude AI* povremeno je dovodilo do prekoračenja petosatne kvote i potrebe za čekanjem, što može utjecati na kontinuitet rada prilikom složenijih zadataka. Osim toga, *Pro* pretplata nije besplatna. Uzevši u obzir da je na predmetu Projektiranje informacijskih sustava godišnje

dosad bilo upisano više od 200 studenata, radi se o ozbiljnijim troškovima. *Oracle Apex*, s druge strane, nudi besplatno okruženje na apex.oracle.com kao i mogućnost pristupanja inicijativi *Oracle Academy*. Još ozbiljnije ograničenje pokazalo se u eksperimentu prebacivanja generirane aplikacije u *Oracle Apex* — dok je uvoz skripti za bazu podataka, testne podatke i REST servise prošao uredno, uvoz SQL skripte same aplikacije generirao je greške koje ni kroz više iteracija s alatom *Claude Code* nisu u cijelosti otklonjene. Taj je rezultat u skladu s uvidima iz recentne literature (Wang i sur., 2025; Xu i sur., 2025), prema kojima zrelost agentskih alata varira ovisno o ciljnoj tehnologiji i domeni, te su oni u pravilu najzreliji u dominantnim ekosustavima poput *Pythona*, dok kod specifičnih komercijalnih platformi poput *Oracle Apexa* još uvijek postoje prepreke. S obzirom na brzi razvoj agentskih modela i samih alata, realno je očekivati da će dio ovih ograničenja u bliskoj budućnosti biti umanjen ili otklonjen.

U nastavku istraživanja planira se provesti isti postupak nad znatno složenijim ERA modelom s više od 15 entiteta, koji odgovara razini složenosti koja se od studenata traži u sklopu izrade timskog projekta na predmetu, kako bi se utvrdilo skaliraju li uočene prednosti i u realnijim poslovnim scenarijima. Isto tako, zanimljivo bi bilo usporediti i druge agentske alate u istom okruženju te istražiti mogućnosti integracije agentskih AI alata u sam nastavni proces kao komplementarnog alata postojećim platformama za malokodno programiranje. U skladu s navedenim, dobivene rezultate ovog rada autor smatra primarno indikativnima za promatrani slučaj jednostavnijeg ERA modela i kontekst izrade prototipa, a ne univerzalno primjenjivima na sve scenarije izrade web aplikacija.

LITERATURA

1. Ge, Y., Mei, L., Duan, Z., Li, T., Zheng, Y., Wang, Y., ... & Cheng, X. (2025). *A Survey of Vibe Coding with Large Language Models*. *arXiv*:2510.12399.
2. Pan, J., Wang, X., Neubig, G., Jaitly, N., Ji, H., Suhr, A., & Zhang, Y. (2024). Training Software Engineering Agents and Verifiers with *SWE-Gym*. arXiv: 2412.21139.
3. Plaat, A., van Duijn, M., Van Stein, N., Preuss, M., van der Putten, P., & Batenburg, K. J. (2025). Agentic large language models, a survey. *Journal of Artificial Intelligence Research*, 84.
4. Tian, M., Wang, Z., Yang, B., Tang, Z., Zhu, K., Dong, H., ... & You, J. (2026). SWE-Bench Mobile: Can Large Language Model Agents Develop Industry-Level Mobile Applications?. arXiv:2602.09540.
5. Vergopoulos, K., Müller, M. N., & Vechev, M. (2025). Automated benchmark generation for repository-level coding tasks. arXiv:2503.07701.
6. Wang, H., Gong, J., Zhang, H., Xu, J., & Wang, Z. (2025). Ai agentic programming: A survey of techniques, challenges, and opportunities. arXiv:2508.11126.
7. Wei, T., Li, T. W., Liu, Z., Ning, X., Yang, Z., Zou, J., ... & He, J. (2026). Agentic reasoning for large language models. arXiv:2601.12538.
8. Xu, J., Deng, K., Li, W., Yu, S., Tang, H., Huang, H., ... & Liu, J. (2025). SWE-Compass: Towards Unified Evaluation of Agentic Coding Abilities for Large Language Models. arXiv:2511.05459.

Summary

Prototyping Web Applications Using an Agentic AI Tool

This paper investigates the possibilities of prototyping web applications using the agentic AI tool Claude Code and compares them with the existing low-code approach in the Oracle Apex environment, which is used in laboratory exercises of the Information Systems Design course at the Faculty of Organization and Informatics. The use case selected was the R1 invoice ERA model with six entities, where only an image file of the ERA model was provided as input to the tool. Based on that, Claude Code autonomously generated a relational database, populated it with test data, and developed a functional web application in the Python/Flask/SQLite technology stack, including CRUD forms for all entities, PDF reports, charts, and REST services returning data in JSON format. The paper presents the prompts used, the selected technologies, additional functionalities implemented through iterative steps, and an experiment of transferring the application to Oracle Apex. The results show that the basic application prototype was developed approximately four times faster than by manual development in Oracle Apex (an estimate based on the author's practical experience and the development time measured in the observed case of a simpler ERA model with six entities), with the option of easily extending it with functionalities that are not trivial to implement in Apex, while the transfer to Oracle Apex revealed problems during the import of the generated application SQL script.

Keywords: *agentic AI; Claude Code; web application prototype; vibe coding; Oracle Apex.*

Autorska prava: ©2026 Autor. Ovaj rad objavljen je u otvorenom pristupu pod licencom Creative Commons Imenovanje – Nekomercijalno 4.0 međunarodna.

Copyright: ©2026 The Author. Open Access: The work is licensed under a Creative Commons Attribution- NonCommercial 4.0. International License.