POS APP

Baza podataka i REST servisi

Upute za uspješno korištenje

Irena Ilišević

1. Sadržaj

2. Opis sustava

Sustav POS App namijenjen je vođenju jednostavne blagajne. Aplikacija omogućava korisniku pregled, unos i izmjenu podatka o artiklima, partnerima, korisnicima, vrstama računa i računima. Program nudi brzo i jednostavno kreiranje i izdavanje računa te njihovo ispisivanje. Sučelje je intuitivno i moderno te vrlo jednostavno za korištenje. Za pristup mogućnostima, korisnik se mora prijaviti te nakon toga može započeti s radom. Osim temeljnih funkcija dodavanja, izmjene i pregleda informacija o artiklima, korisnicima, partnerima i računima, korisniku su dostupne i opcije pretraživanja i sortiranja podataka. Osim toga, prisutan je i prikaz jednostavne statistike.

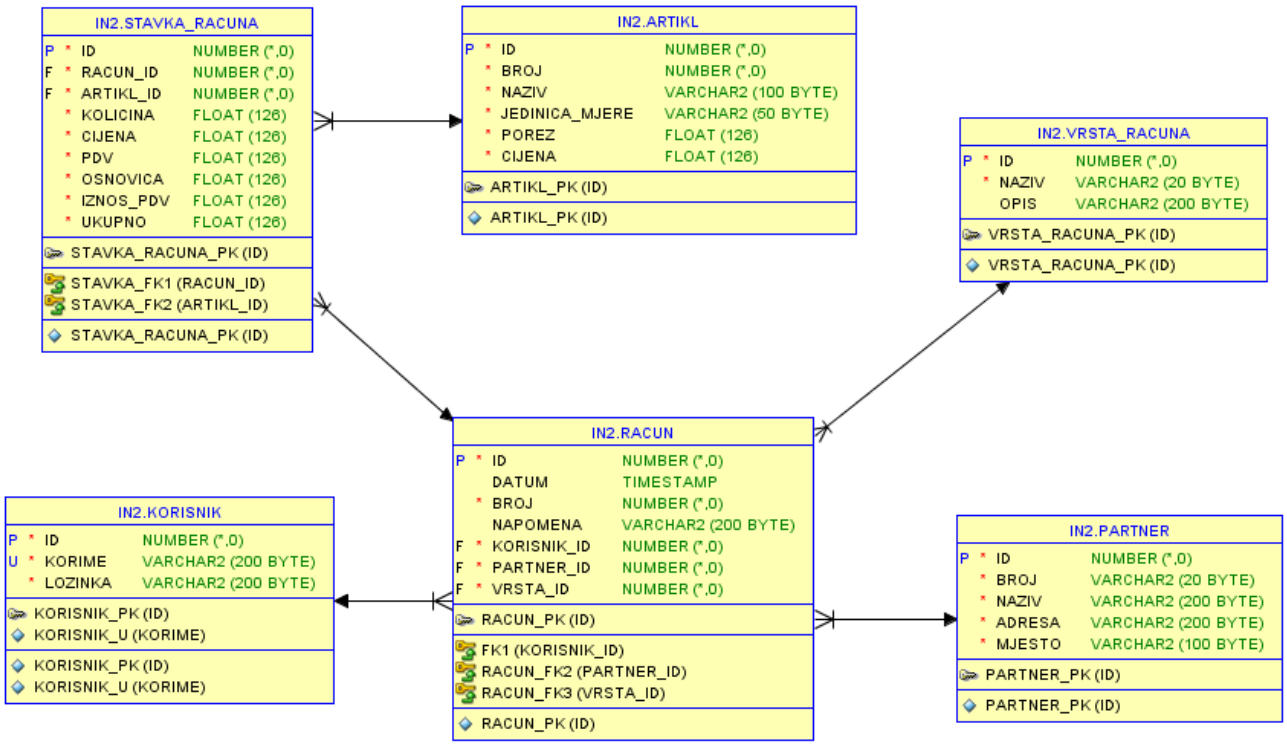
3. Baza podataka

i. Opis domene

Baza podataka za potrebe upravljanje jednostavnom blagajnom sastoji se od nekoliko entiteta. Domena koju baza treba modelirati podrazumijeva postojanje artikala – proizvoda koji će se prodavati, korisnika – osobe koje će se moći prijaviti u sustav te ga koristiti, partnera – partneri s kojima poduzeće surađuje, vrste računa – vrste računa koje su podržane za izdavanje kroz sustav, računa – računi i stavke računa izdanih kroz implementirani sustav. Sama baza podataka mora omogućiti pohranu, dodavanje, uređivanje i brisanje podataka o ovim entitetima. Za pravilno funkcioniranje sustava za izdavanje računa potrebno je imati valjano unesene artikle, partnere i vrste računa. Sam sustav račune kreira na temelju postojećih zapisa o artiklima, partnerima i vrstama računa iz baze podataka. Za svaki od entiteta važno je zabilježiti neke od atributa, tako na primjer, za svaki artikl je važan njegov naziv, cijena, iznos PDV-a i slično. Ove informacije ključne su za izdavanje računa.

ii. ERA model

Modeliranjem baze podataka – definiranjem entiteta, atributa i veza dobiva se logički model podataka (ERA model). Sama baza podataka implementirana je u sustavu za upravljanje bazama podataka Oracle 19c, a pripadajući model prikazan je na sljedećoj slici:



iii. Relacijska schema

U prikazu relacijske scheme primarni ključevi su podebljani (bold), a vanjski su ukošeni (italic):

ARTIKL (**ID**, BROJ, NAZIV, JEDINICA_MJERE, POREZ, CIJENA);
VRSTA_RACUNA (**ID**, NAZIV, OPIS);
PARTNER (**ID**, BROJ, NAZIV, ADRESA, MJESTO);
KORISNIK (**ID**, KORIME, LOZINKA);
RACUN (**ID**, DATUM, BROJ, NAPOMENA, *KORISNIK_ID, PARTNER_ID, VRSTA_ID*);
STAVKA_RACUNA (**ID**, *RACUN_ID, ARTIKL_ID,* KOLICINA, CIJENA, PDV, OSNOVICA, IZNOS_PDV, UKUPNO);

iv. Opis entiteta i atributa

U bazi podataka temeljenoj na prikazanoj relacijskoj schemi i ERA modelu jaki entiteti su Artikl, Korisnik, Partner, Vrsta_racuna i Racun, dok je slabi entitet Stavka_racuna. U tablicama koje slijede opisani su entiteti, njihovi atributi te je naveden po jedan primjer zapisa za svaki od entiteta.

ARTIKL

Opis entiteta	Tablica za spremanje podataka o artiklima koji su dostupni za prodaju putem POS sustava. Opisuje svaki pojedini artikl na nekoliko razina.
Atributi	ID – NUMBER (identity column), jedinstveni identifikator artikla, primarni ključ relacije, obavezno polje BROJ – NUMBER, šifra artikla, obavezno polje NAZIV – VARCHAR2(100), naziv artikla, obavezno polje JEDINICA_MJERE – VARCHAR2(50), jedinica u kojoj se mjeri artikl (oznaka jedinice mjere), obavezno polje POREZ – FLOAT, iznos poreza na artikl u %, obavezno polje CIJENA – FLOAT, jedinična cijena artikla, obavezno polje
Primjer	(1, 123456, 'Tipkovnica', 'komad', 10, 25)

KORISNIK

Opis entiteta	Tablica za spremanje podataka o korisnicima koji mogu pristupiti sustavu za izdavanje računa i koji mogu izdati neki račun.
Atributi	ID – NUMBER (identity column), jedinstveni identifikator korisnika, primarni ključ relacije, obavezno polje KORIME – VARCHAR2(200), korisničko ime korisnika za pristup sustavu, obavezno polje LOZINKA – VARCHAR2(200), lozinka korisnika za pristup sustavu u kriptiranom obliku, obavezno polje
Primjer	(1, 'admin', '3B143023D5B984F9601FF760AE12DA54')

PARTNER

Opis entiteta	Tablica za spremanje podataka o partnerima koji surađuju s poduzećem.
Atributi	ID – NUMBER (identity column), jedinstveni identifikator partnera, primarni ključ relacije, obavezno polje BROJ – VARCHAR2(20), kontakt broj partnera, obavezno polje NAZIV – VARCHAR2(200), naziv partnera, obavezno polje ADRESA – VARCHAR2(200), ulica i broj (adresa) partnera, obavezno polje MJESTO – VARCHAR2(100), mjesto partnera, obavezno polje
Primjer	(1,'91-892-195', 'Farrell and Sons', '80276 Oxford Circle', 'Xuetian')

VRSTA_RACUNA

Opis entiteta	Tablica za spremanje podataka o vrstama računa koji se mogu izdavati korištenjem sustava.
Atributi	ID – NUMBER (identity column), jedinstveni identifikator vrste računa, primarni ključ relacije, obavezno polje NAZIV – VARCHAR2(20), naziv vrste računa, obavezno polje OPIS – VARCHAR2(200), opis vrste računa
Primjer	(1,'R1' , 'Račun koji se izdaje poduzećima kako bi mogli odbiti PDV')

RACUN

Opis entiteta	Tablica za spremanje podataka o zaglavlju računa.
Atributi	ID – NUMBER (identity column), jedinstveni identifikator računa, primarni ključ relacije, obavezno polje DATUM – TIMESTAMP, datum i vrijeme izdavanja računa, obavezno polje BROJ – NUMBER, broj računa, obavezno polje NAPOMENA – VARCHAR2(200), napomena uz račun (ako je ima) KORISNIK_ID – NUMBER, oznaka korisnika koji je izdao račun, vanjski ključ na relaciju KORISNIK, obavezno polje PARTNER_ID – NUMBER, oznaka partnera, vanjski ključ na relaciju PARTNER, obavezno polje VRSTA_ID – NUMBER, oznaka vrste računa, vanjski ključ na relaciju VRSTA_RACUNA, obavezno polje
Primjer	(1, '19-MAR-21 12.00.00.00 AM' ,1111, 'Plaćanje na rate', 1, 1, 1)

STAVKA_RACUNA

Opis entiteta	Tablica za spremanje podataka o stavkama računa.
Atributi	ID – NUMBER (identity column), jedinstveni identifikator stavke računa, primarni ključ relacije, obavezno polje RACUN_ID – NUMBER, oznaka računa na koji se stavka odnosi, vanjski ključ na relaciju RACUN, obavezno polje ARTIKL_ID – NUMBER, oznaka artikla koji je stavka, vanjski ključ na relaciju ARTIKL, obavezno polje KOLICINA – FLOAT, količina artikla, obavezno polje CIJENA – FLOAT, jedinična cijena artikla, obavezno polje PDV – FLOAT, porez na artikl u %, obavezno polje OSNOVICA – FLOAT, iznos osnovice stavke (količina*jedinična cijena), obavezno polje IZNOS_PDV – FLOAT, iznos poreza na stavku ((osnovica*pdv)/100), obavezno polje UKUPNO – FLOAT, ukupni iznos stavke (osnovica + iznos pdv-a), obavezno polje
Primjer	(1, 1, 1, 2, 66.7, 20, 133.4, 26.68, 160.08)

v. Veze između entiteta

U sljedećoj tablici opisane su veze između entiteta.

Veza između entiteta	*Opis veze*
RACUN - KORISNIK	Jedan račun može izdati samo jedan korisnik, jedan korisnik može izdati više računa
RACUN - PARTNER	Na jednom računu može se nalaziti samo jedan partner, jedan partner može biti na više računa
RACUN – VRSTA_RACUNA	Jedan račun može biti samo jedne vrste, jedne vrste računa može biti više računa
STAVKA_RACUNA - ARTIKL	Jedna stavka računa može imati samo jedan artikl, jedan artikl može biti na više stavki
STAVKA_RACUNA - RACUN	Jedna stavka može biti na jednom računu, jedan račun može imati više stavki

vi. Funkcije

Baza podataka ima dvije kreirane funkcije. To su funkcije: AUTHENTICATE_USER i HASH_PASSWORD.

Zadaća funkcije AUTHENTICATE_USER je provjeriti korisničko ime i dobivenu lozinku. Funkcija provjerava postojanje korisnika u bazi podataka te vraća rezultat ovisno o tome je li korisnik pronađen ili ne. Funkcija prima dva parametra: korisničko ime i lozinku, a može vratiti jednu od 4 vrijednosti:

0 – uspješna autentifikacija
1 – nepoznato korisničko ime
2 – pogrešna lozinka
3 – nepoznata greška

Funkcija HASH_PASSWORD služi za kreiranje hash-a unesene lozinke. Funkcija prima parametre korisničko ime i lozinku, a vraća lozinku u kriptiranom obliku.

vii. Okidači

U bazi podataka kreirana su 3 okidača: BI_USERS, BU_USERS, RACUN_TRG.

Okidač BI_USERS prije umetanja u tablicu KORISNIK poziva funkciju HASH_PASSWORD za dobivanje kriptirane verzije unesene lozinke i sprema kriptiranu lozinku u tablicu.

Okidač BU_USERS prije ažuriranja reda u tablici KORISNIK poziva funkciju HASH_PASSWORD za dobivanje kriptirane verzije izmijenjene lozinke i sprema ju u tablicu.

Okidač RACUN_TRG služi za generiranje sljedećeg broja računa prije umetanja u tablicu RACUN.

4. REST servisi

Resource Template	HTTP Method	Parametri	SQL upit	Povratna vrijednost
Artikli: http://localhost:8181/ords/in2/api/artikli	GET	id - id željenog artikla	select * from artikl WHERE :id IS NULL OR id = :id	Vraća sve artikle ukoliko nije dobiven parametar id. Ukoliko je dobiven id vraća podatke o artiklu čiji je id dobiven. Ukoliko artikl s dobivenim id-jem ne postoji vraća prazno polje.

	POST	broj – broj novog artikla,
		naziv – naziv novog artikla,
		jed_mjere – jedinica mjere novog artikla,
		porez – stopa poreza novog artikla,
		cijena – jedinična cijena novog artikla
			begin
			--
			insert into artikl(
			broj,
			naziv,
			jedinica_mjere,
			porez,
			cijena
			) values(
			:broj,
			:naziv,
			:jed_mjere,
			:porez,
			:cijena
			);
			--
			:status := 200; --created
			--
			exception when others then
			:status := 400; --error
			:errmsg := sqlerrm;
			end;
				Vraća status 200 ukoliko je artikl uspješno kreiran.
				Inače vraća status 400 i tekst pogreške SQL upita.

	PUT	id – id artikla koji se želi ažurirati, broj – broj artikla,	begin if :id is not null then	Vraća status 200 ukoliko je ažuriranje uspjelo.

		naziv – naziv artikla,
		jed_mjere – jedinica mjere artikla,
		porez - stopa poreza na artikl,
		cijena – jedinična cijena artikla
			for i in (select * from artikl where id=:id) loop
			update artikl set
			broj= nvl(:broj, i.broj),
			naziv=nvl(:naziv, i.naziv),
			jedinica_mjere=nvl(:jed_mjere, i.jedinica_mjere), porez=nvl(:porez, i.porez), cijena=nvl(:cijena, i.cijena)
			where id=:id;
			end loop;
			--
			if sql%found then
			:status :=200; --success
			else
			:status :=400;
			:errmsg := 'not found';
			end if;
			else
			:status :=400;
			:errmsg :='id must be specified';
			end if;
			exception
			when others then
			:status :=400;
			:errmsg := sqlerrm;
			end;
				Ako nije specificiran neki od ulaznih parametara, upit ostavlja u tablici staru vrijednost atributa.
				Vraća status 400 i pogrešku 'not found' ukoliko artikl s danim id-jem ne postoji u bazi podataka.
				Vraća status 400 i pogrešku 'id must be specified' ukoliko nije dobiven parametar id.
				Vraća 400 i pogrešku SQL upita ukoliko je neka druga greška u pitanju.

Korisnici: http://localhost:8181/ords/in2/api/korisnici	GET	id – id željenog korisnika	select * from korisnik WHERE :id IS NULL OR id = :id	Vraća sve korisnike ukoliko parametar id nije specificiran. Ukoliko je parametar id specificiran vraća podatke o korisniku s dobivenim id-jem. U svakom drugom slučaju vraća prazno polje.
	POST	korime – korisničko ime novog korisnika, lozinka – lozinka novog korisnika	begin -- insert into korisnik(korime, lozinka) values(	Vraća status 200 ukoliko je korisnik uspješno kreiran. Vraća status 400 i pogrešku SQL upita ukoliko je došlo do pogreške.

			:korime, :lozinka); -- :status := 200; --created -- exception when others then :status := 400; --error :errmsg := sqlerrm; end;	

	PUT	id – id korisnika koji se želi ažurirati,
		korime – korisničko ime,
		lozinka – lozinka korisnika
			begin
			if :id is not null then
			for i in (select * from korisnik where id=:id) loop
			update korisnik set
			korime= nvl(:korime, i.korime),
			lozinka=nvl(:lozinka, i.lozinka)
			where id=:id;
			end loop;
			--
			if sql%found then
			:status :=200; --success
			else
			:status :=400;
			:errmsg := 'not found';
			end if;
			else
			:status :=400;
			:errmsg :='id must be specified';
			end if;
			exception
			when others then
			:status :=400;
			:errmsg := sqlerrm;
			end;
				Vraća status 200 ukoliko je ažuriranje uspjelo.
				Ako nije specificiran neki od ulaznih parametara upit ostavlja u tablici staru vrijednost atributa.
				Vraća status 400 i pogrešku 'not found' ukoliko korisnik s danim id-jem ne postoji u bazi podataka.
				Vraća status 400 i pogrešku 'id must be specified' ukoliko nije dobiven parametar id.
				Vraća 400 i pogrešku SQL upita ukoliko je neka druga greška u pitanju.

Partner: http://localhost:8181/ords/in2/api/partner	GET	id – id željenog partnera	select * from partner WHERE :id IS NULL OR id = :id	Vraća sve partnere ukoliko parametar id nije specificiran. Ukoliko je parametar id specificiran vraća podatke o partneru s dobivenim id-jem. U svakom drugom slučaju vraća prazno polje.

	POST	broj – kontakt broj novog partnera,
		naziv – naziv novog partnera,
		adresa – adresa novog partnera,
		mjesto – mjesto novog partnera
			begin
			--
			insert into partner(
			broj,
			naziv,
			adresa,
			mjesto
			
			) values(
			:broj,
			:naziv,
			:adresa,
			:mjesto
			
			);
			--
			:status := 200; --created
			--
			exception when others then
			:status := 400; --error
			:errmsg := sqlerrm;
			end;
				Vraća status 200 ukoliko je partner uspješno kreiran.
				Vraća status 400 i pogrešku SQL upita ukoliko je došlo do pogreške.

	PUT	id – id partnera koji se želi ažurirati,
		broj – kontakt broj,
		naziv – naziv partnera,
		mjesto – mjesto partnera,
		adresa – adresa partnera
			begin
			if :id is not null then
			for i in (select * from partner where id=:id) loop
			update partner set
			broj= nvl(:broj, i.broj),
			naziv=nvl(:naziv, i.naziv),
			mjesto=nvl(:mjesto, i.mjesto),
			adresa=nvl(:adresa, i.adresa)
			where id=:id;
				Vraća status 200 ukoliko je ažuriranje uspjelo.
				Ako nije specificiran neki od ulaznih parametara upit ostavlja u tablici staru vrijednost atributa.
				Vraća status 400 i pogrešku 'not found' ukoliko partner s danim id-jem ne postoji u bazi podataka.
				Vraća status 400 i pogrešku 'id must be specified' ukoliko nije dobiven parametar id.

			end loop;
			--
			if sql%found then
			:status :=200; --success
			else
			:status :=400;
			:errmsg := 'not found';
			end if;
			else
			:status :=400;
			:errmsg :='id must be specified';
			end if;
			exception
			when others then
			:status :=400;
			:errmsg := sqlerrm;
			end;
				Vraća 400 i pogrešku SQL upita ukoliko je neka druga greška u pitanju.

Racuni: http://localhost:8181/ords/in2/api/racuni	GET	id – id željenog računa	select racun.id racunRC, racun.datum datumRC, racun.broj brojRC, racun.napomena napomenaRC, partner.naziv partnerRC, korisnik.korime korisnikRC, vrsta_racuna.naziv vrstaRC from racun inner join korisnik on racun.korisnik_id=korisnik.id inner join partner on racun.partner_id=partner.id inner join vrsta_racuna on racun.vrsta_id=vrsta_racuna.id WHERE :id IS NULL OR racun.id = :id	Vraća sve račune ukoliko parametar id nije specificiran. Ukoliko je parametar id specificiran vraća podatke o računu s dobivenim id-jem. U svakom drugom slučaju vraća prazno polje.
	POST	datum – datum novog računa,	begin INSERT INTO racun (datum, napomena, korisnik_id, partner_id,	Vraća id novokreiranog računa ukoliko je račun uspješno kreiran. U svakom drugom slučaju vraća prazno polje.

		napomena – napomena novog računa, korisnik – id korisnika koji je kreirao račun, partner – id partnera novog računa vrsta – id vrste novog računa	vrsta_id) VALUES (:datum, :napomena, :korisnik, :partner, :vrsta) RETURNING id INTO :racunID; end;	
Stavke: http://localhost:8181/ords/in2/api/stavke	GET	id – id računa čije se stavke žele dohvatiti	select s.*, a.naziv from stavka_racuna s, artikl a WHERE s.racun_id = :id and s.artikl_id=a.id	Vraća sve stavke računa s dobivenim id-jem. U svakom drugom slučaju vraća prazno polje.
	POST	racun – id računa, artikl – id artikla, kolicina – količina stavke, cijena – jedinična cijena stavke, pdv – porez na artikl, osnovica – osnovica stavke, iznospdv – iznos pdv-a na stavku, ukupno – ukupni iznos stavke	insert into stavka_racuna(racun_id, artikl_id, kolicina, cijena, pdv, osnovica, iznos_pdv, ukupno) values (:racun, :artikl, :kolicina, :cijena, :pdv, :osnovica, :iznospdv, :ukupno)	Vraća status 200 ukoliko je stavka uspješno kreirana. U svakom drugom slučaju vraća pogrešku.
VrsteRacuna: http://localhost:8181/ords/in2/api/vrsteRacuna	GET	id – id željene vrste računa	select * from vrsta_racuna WHERE :id IS NULL OR id = :id	Vraća sve vrste računa ukoliko parametar id nije specificiran. Ukoliko je parametar id specificiran vraća podatke o vrsti računa s dobivenim id-jem. U svakom drugom slučaju vraća prazno polje.
	POST	naziv – naziv nove vrste računa, opis – opis nove vrste računa	begin -- insert into vrsta_racuna(naziv,	Vraća status 200 ukoliko je vrsta računa uspješno kreirana. Vraća status 400 i pogrešku SQL upita ukoliko je došlo do pogreške.

			opis) values(:naziv, :opis); -- :status := 200; --created -- exception when others then :status := 400; --error :errmsg := sqlerrm; end;	

	PUT	id – id vrste računa koja se želi ažurirati,
		naziv – naziv vrste računa,
		opis – opis vrste računa
			begin
			if :id is not null then
			for i in (select * from vrsta_racuna where id=:id) loop
			update vrsta_racuna set
			naziv= nvl(:naziv, i.naziv),
			opis=nvl(:opis, i.opis)
			where id=:id;
			end loop;
			--
			if sql%found then
			:status :=200; --success
			else
			:status :=400;
			:errmsg := 'not found';
			end if;
			else
			:status :=400;
			:errmsg :='id must be specified';
			end if;
			exception
			when others then
			:status :=400;
			:errmsg := sqlerrm;
			end;
				Vraća status 200 ukoliko je ažuriranje uspjelo.
				Ako nije specificiran neki od ulaznih parametara upit ostavlja u tablici staru vrijednost atributa.
				Vraća status 400 i pogrešku 'not found' ukoliko vrsta računa s danim id-jem ne postoji u bazi podataka.
				Vraća status 400 i pogrešku 'id must be specified' ukoliko nije dobiven parametar id.
				Vraća 400 i pogrešku SQL upita ukoliko je neka druga greška u pitanju.