

**SVEUČILIŠTE U ZAGREBU
FAKULTET ORGANIZACIJE I INFORMATIKE
VARAŽDIN**

Vanja Mačković

SKLADIŠTE RAČUNALNIH KOMPONENTI

PROJEKT

Varaždin, 2021.

SVEUČILIŠTE U ZAGREBU
FAKULTET ORGANIZACIJE I INFORMATIKE
V A R A Ž D I N

Vanja Mačković

Matični broj: 45919/17–R

Studij: Organizacija poslovnih sustava

SKLADIŠTE RAČUNALNIH KOMPONENTI

PROJEKT

Mentor:

Dr. sc. Bogdan Okreša Đurić

Varaždin, Siječanj 2021.

Vanja Mačković

Izjava o izvornosti

Izjavljujem da je moj projekt izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

Autor potvrdio prihvaćanjem odredbi u sustavu FOI-radovi

Sažetak

Tema ovog projekta je bila stvoriti aplikaciju koja će automatizirati dio poslova koji se odvijaju u skladištu te pružiti uvid u podatke koji će pomoći u kontroli skladišta kao što su povijest stanja ili narudžbi. To je ostvareno pomoću aktivnih i temporalnih komponenti baze podataka. Za primjer jednog takvog skladišta koristio sam skladište računalnih komponenti. Pošto su danas komponente dosta raznovrsne onda postoji i potražnja za dobrom kontrolom svih tih komponenti, a uz to sam i zainteresiran za to područje. Rezultat projekta je aplikacija napravljena u C-Sharp jeziku u Visual Studio alatu koja se povezuje s bazom podataka koja je kreirana pomoću pgAdmin4 alata te uređivana pomoću Navicat 15 alata. Rad na projektu je bio zanimljiv te sam poneku stvar i naučio.

Ključne riječi: Skladište, Komponente, DB, pgAdmin4, C-Sharp, Navicat, Aktivne, Temporalne, PostgreSQL

Sadržaj

1. Opis aplikacijske domene	1
2. Teorijski uvod	2
2.1. Aktivne baze podataka	2
2.2. Temporalne baze podataka	2
3. Model baze podataka	3
3.1. ERA model	3
4. Implementacija	5
4.1. Korišteni alati za izradu	5
4.1.1. pgAdmin4	5
4.1.2. Navicat 15	6
4.1.3. Visual Studio	7
4.2. Izrada baze podataka	7
4.3. Okidači	11
4.3.1. Evidencija ažuriranja količine komponenti	11
4.3.2. Evidencija brisanja komponente sa skladišta	12
4.3.3. Evidencija nove komponente na skladište	12
4.3.4. Stvaranje narudžbe	13
4.3.5. Ažuriranje stanja skladišta sa dostavljenom komponentom	13
4.3.6. Evidentiranje izvršene narudžbe	14
4.4. Aplikacija	14
4.4.1. Spajanje sa bazom podataka	15
4.4.2. Dohvaćanje podataka iz baze	15
4.4.3. Unos podataka u bazu	16
4.4.4. Ažuriranje baze podataka	16
4.4.5. Brisanje podataka u bazi	17
5. Primjeri korištenja	18
5.1. Glavni izbornik	18
5.2. Stanje skladišta	18
5.3. Nabava komponenti	19
5.4. Unos komponenti	20
5.5. Unos karakteristika	21
5.6. Pregled svih komponenti	22

6. Zaključak	23
Popis literature	24
Popis slika	25
Popis popis tablica	26
1. Prilog 1	27
2. Prilog 2	28

1. Opis aplikacijske domene

Tema ovog projekta je stvaranje aplikacije za vođenje statistike skladišta i planiranje zaliha. Kroz rad je objašnjeno na koji način je to postignuto, teorijski i praktično. U radu su korištene komponente aktivnih i temporalnih baza podataka u sklopu s aplikacijom koja prikazuje sve napravljene mogućnosti i omogućuje upravljanje bazom podataka pomoću istih. Da bi sve to funkcioniralo mora se napraviti nekoliko nužnih stvari. Kreirana je baza podataka u pgAdmin4 alatu gdje je postavljena lokacija baze i neke njezine osnovne postavke koje dopuštaju povezivanje s aplikacijom. Ono što je također korišteno je alat Navicat 15, koji kada se poveže sa pgAdmin4 bazom, omogućava uređivanje te iste baze podataka. Sljedeće što je potrebno je napraviti aplikaciju, za što je korišten Visual Studio i jezik C-Sharp.

2. Teorijski uvod

Izrada ovog projekta zahtijevala je korištenje aktivnih i temporalnih komponenti baza podataka. U ovom projektu je pomoću tih komponenti napravljeno nekoliko ključnih dijelova aplikacija pomoću kojih će se onda lakše upravljati skladištem. Aktivni dio je zadužen za automatizaciju aplikacije. Ovisno o radnjama koje se rade u aplikaciji kao što je primjerice ažuriranje stanja neke računalne komponente na skladištu, u pozadini će se paralelno odrađivati neki drugi procesi. To je ostvareno pomoću okidača (eng. trigger). Drugi dio je korištenje temporalnih komponenti. Navedene komponente su se koristile za prikaz stanja skladišta u nekom drugom vremenu od trenutnog. U projektu su se primarno koristile za prikaz povijesti stanja skladišta.

2.1. Aktivne baze podataka

Aktivna baza podataka je baza podataka koja se sastoji od skupa okidača. Okidači su dijelovi koda koji se aktiviraju u samo određenim preddefiniranim slučajevima i izvršavaju točno preddefinirane akcije [1]. Moguće aktivirati više okidača odjednom direktno ili indirektno tako što se definiraju da se aktiviraju na isti uvjet. Direktno kada se akcijom korisnika postigne određeni uvjet, a indirektno ako se izvršavanjem naredbi jednog okidača aktivira drugi okidač. Indirektno aktivirani okidači i dalje izvršavaju samo ono za što su stvoreni, nije važno rezultatom čega je aktiviran.

Okidači ako su dobro postavljeni i definirani mogu biti samo prednost u odnosu na to da ih uopće nema. Olakšavaju upravljanje bazom podataka i smanjuju vjerojatnost pogreške jer korisnik ovim putem obavlja manje radnji, a time i manje grešaka može napraviti.

Jedna od mana ovog tipa je održavanje i razumijevanje baze podataka. Pošto se okidači mogu postaviti za široki spektar operacija nad bazom podataka, to dopušta razne funkcionalnosti koje onda razumije nekolicina ljudi te je potrebno dobro poznavanje baze za buduće radnje nad bazom.

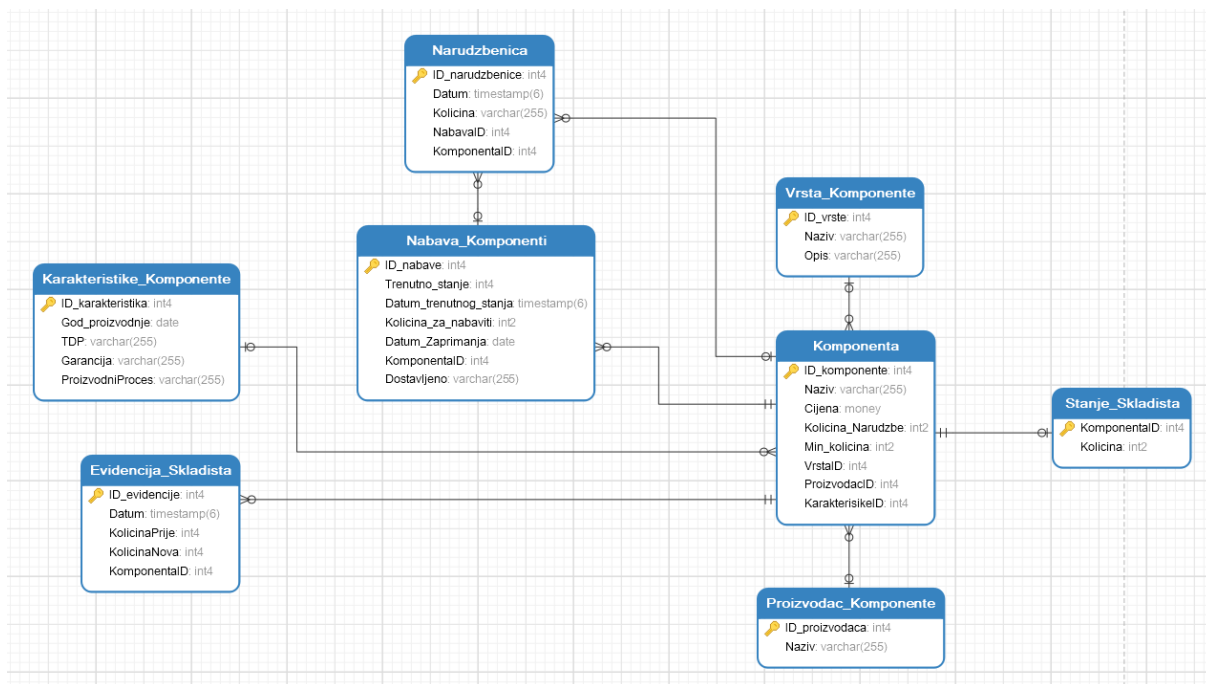
2.2. Temporalne baze podataka

Temporalna baza podataka je baza podataka koja ima za mogućnost skupljanje vremenskih podataka. Mogu se bilježiti vremenski podaci o prošlosti, budućnosti ili sadašnjosti [2]. Takve baze podataka primjerice omogućuju uvid u prijašnje stanje varijabli ili uvid u stanje varijabli tijekom nekog razdoblja. Ovakav tip uporabe temporalnih baza podataka će biti korišteni i u ovom projektu. Aplikacija nudi mogućnost uvida u povijest stanja pojedine komponente na skladištu te također uvid u promjene stanja tijekom nekog razdoblja.

3. Model baze podataka

3.1. ERA model

Baza podataka potrebna za skladište računalnih komponenti mora sadržavati određene tablice. Ispod se nalazi slika 1 koja prikazuje popis kreiranih tablica i njihove atribute. Ispod slike su objašnjenja za svaku tablicu. ERA model napravljen je u alatu Navicat 15 koji prema tablicama i vezama između njih kreira točan model baze podataka.



Slika 1: ERA model (Vlastita izrada)

- **Komponenta** - Tablica Komponenta je glavna tablica baze podataka koja sadrži sve komponente koje se nalaze na skladištu. Povezana je sa svim ostalim tablicama iz kojih onda dohvaća najbitnije potrebne informacije.
- **Vrsta Komponente** - Pod pojmom računalna komponenta spada više stavaka kao što su procesor, grafička kartica, napajanje, memorija, tvrdi disk te je zbog jednostavnosti kod unosa komponenata na skladište stvorena ova tablica koja korisniku olakšava identifikaciju o kakvoj se vrsti komponente točno radi.
- **Stanje Skladista** - Ova tablica sadrži trenutnu količinu svake računalne komponente koja se nalazi na skladištu.
- **Proizvodac Komponente** - Uvijek postoji mogućnost da više proizvođača proizvodi istu vrstu računalne komponente te je zbog toga potrebna tablica koja sadrži samo proizvođače da ne bi došlo do navođenja krivog proizvođača.
- **Nabava Komponenti** - Tablica nabava služi za popis komponenti koje treba naručiti. Kada se količina komponente na skladištu smanji ispod minimalne količine, a da nije mak-

nuta potpuno iz skladišta, odnosno da se uopće neće više naručivati onda se isključivo pomoću okidača stvara stavka u ovoj tablici.

- **Karakteristike Komponente** - Komponente mogu imati iste karakteristike te je zbog jednostavnosti unosa novih komponenata napravljena tablica koja sadrži sve karakteristike do sada unesenih komponenti. Također služi i za unos novih karakteristika kada dođe novi proizvod koji do sada nije bio na skladištu.
- **Evidencija Skladista** - Tablica služi za zapis svih promjena na skladištu, odnosno svih promjena nad tablicom "Stanje skladista". To uključuje dodavanje novih komponenata na skladište, mijenjanje postojeće količine skladišta i uklanjanje komponente sa skladišta. Jako važna tablica koja omogućuje kontrolu skladišta ali i analizu promjena.
- **Narudzbenica** - Tablica sadrži sve narudžbe koje su trenutno u tijeku. Podatke dobiva iz tablice "Nabava komponenti" kada napravimo narudžbu potrebnih komponenata.

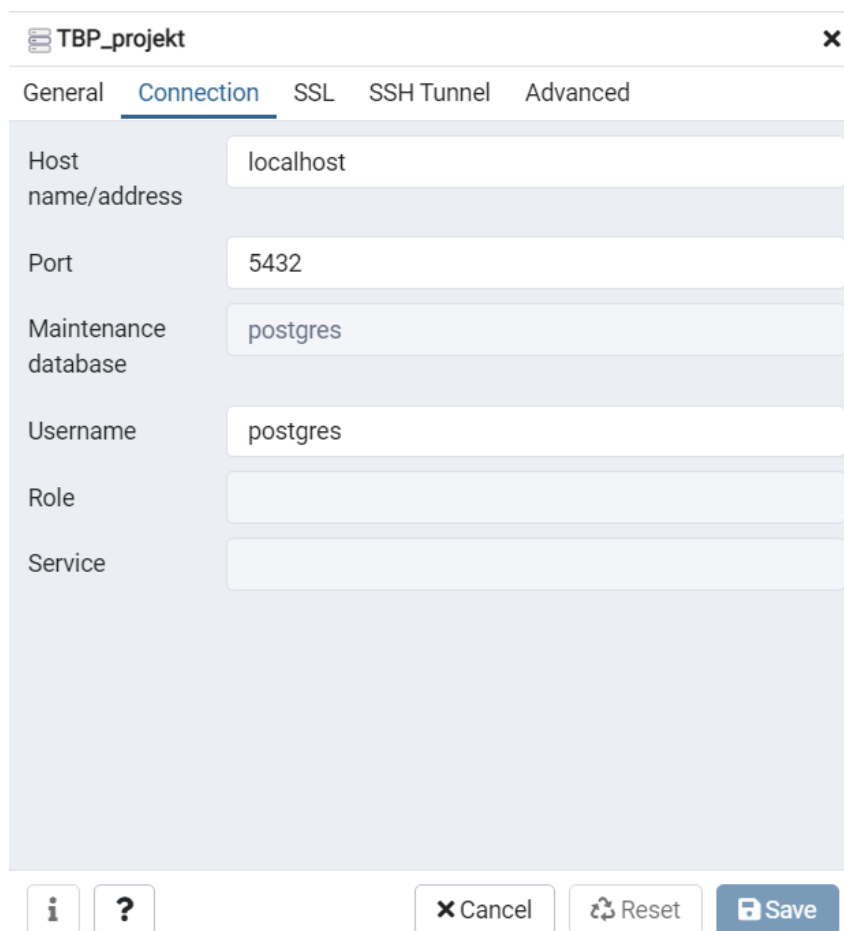
4. Implementacija

Implementacija projekta zahtijevala je nekoliko alata koji su u poglavlju ispod navedeni. Redoslijed implementacije je bio takav da je prvo trebalo kreirati server na kojem će se nalaziti baza podataka. Nakon toga bilo je potrebno kreirati bazu podataka, sve potrebne tablice, sekvence za primarne ključeve, okidače te napuniti tablice nekim testnim podacima. Sljedeći korak je bio napraviti aplikaciju gdje su napravljene sve potrebne forme i napravljena je poveznica na bazu podataka.

4.1. Korišteni alati za izradu

4.1.1. pgAdmin4

Alat koji je korišten prilikom kreiranja baze podataka je pgAdmin4 [3]. Pomoću njega je napravljen lokalni server za bazu podataka. Prilikom postavljanja servera lokalne baze podataka mora se upisati adresa, u ovom slučaju localhost koji predstavlja naše računalo kao host i port na kojem će se promet provoditi, u ovom slučaju je to port 5432. Server je nazvan "TBP projekt".

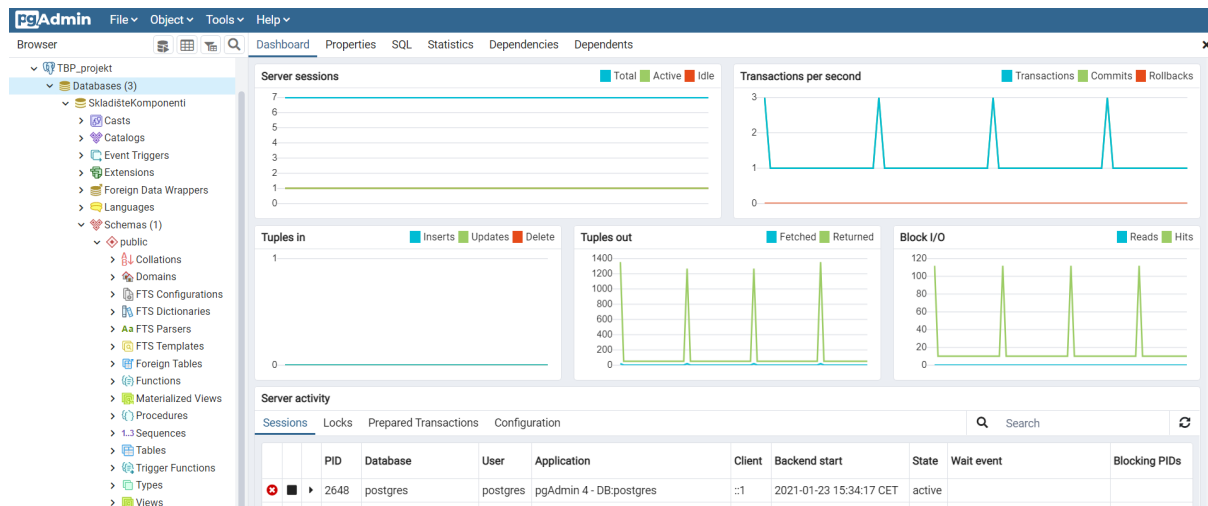


The screenshot shows the pgAdmin4 interface for creating a new connection. The window title is 'TBP_projekt'. The 'Connection' tab is selected, showing fields for Host name/address (localhost), Port (5432), Maintenance database (postgres), Username (postgres), Role, and Service. At the bottom, there are buttons for information, help, Cancel, Reset, and Save.

Field	Value
Host name/address	localhost
Port	5432
Maintenance database	postgres
Username	postgres
Role	
Service	

Slika 2: Postavke veze (Vlastita izrada)

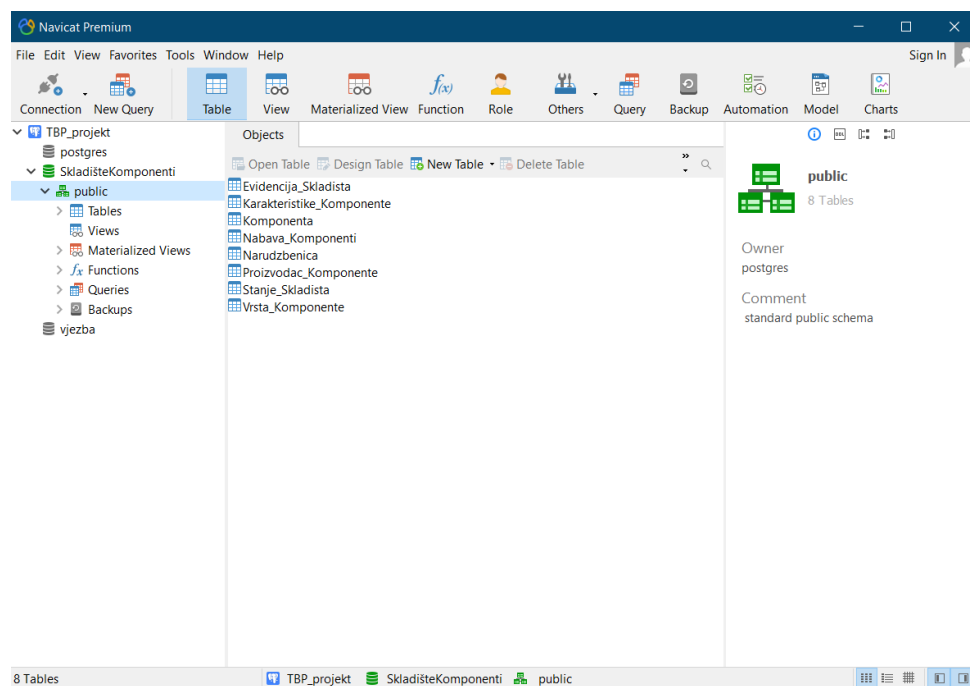
Nakon što je server napravljen, može se dodati nova baza podataka koja je nazvana "SkladišteKomponenti". Dodavanjem baze podataka dolazi se na sljedeći prozor koji govori o svim bitnijim informacijama trenutno označene baze podataka kao što su broj trenutnih sesija, broj transakcija po sekundi te tko sve trenutno pristupa bazi podataka.



Slika 3: Informacije o bazi (Vlastita izrada)

4.1.2. Navicat 15

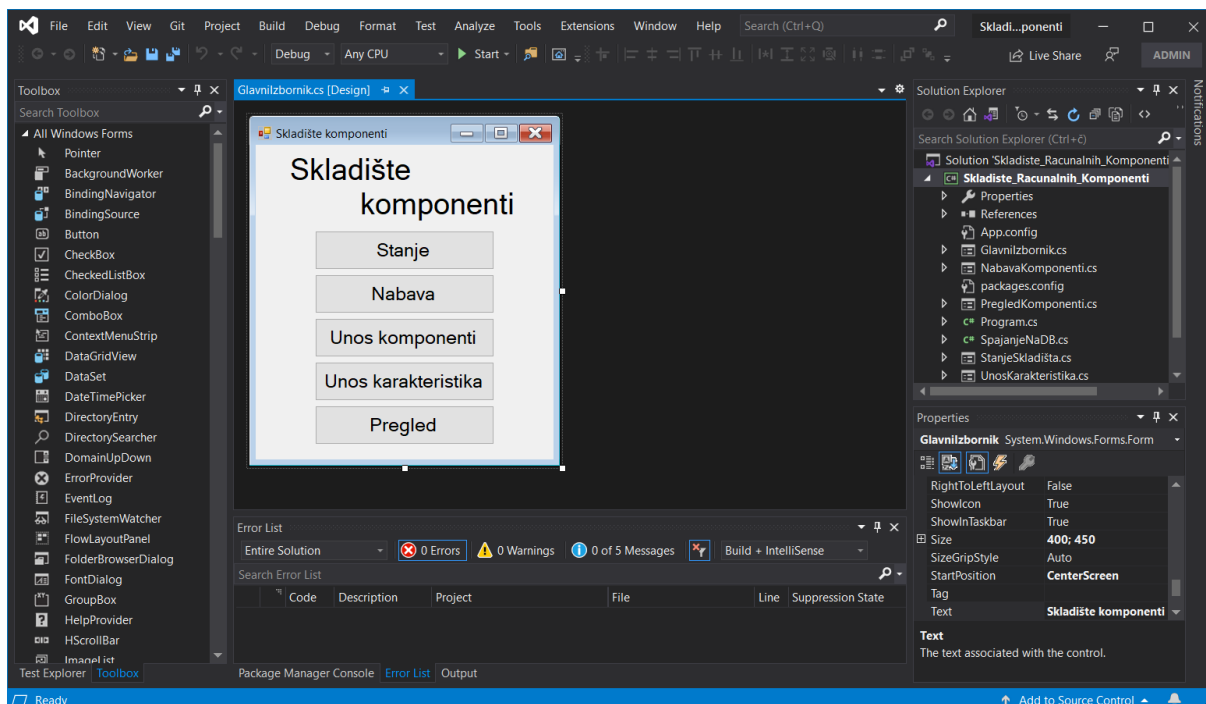
Iako je sve moguće napraviti u pgAdmin4 alatu, ipak se koristio i alat Navicat 15 [4]. Navicat 15 je olakšao kontroliranje baze podataka za potrebe ovog projekta zbog svoje intuitivnosti i lakoće uporabe. Alat je korišten za kreiranje tablica, okidača, upita itd. Na slici ispod može se vidjeti pogled na listu tablica baze "SkladišteKomponenti".



Slika 4: Prikaz tablica u alatu Navicat 15 (Vlastita izrada)

4.1.3. Visual Studio

Aplikacija koja će povezati bazu podataka i korisnika je napravljena u C-Sharp jeziku, u Visual Studio alatu. Visual Studio je intuitivan alat za stvaranje Windows Forms aplikacija [5]. Razlog zbog čega je korišten baš taj alat je prijašnje rukovanje s alatom na jednom od kolegija na fakultetu. Da bi se aplikacija mogla povezati s bazom podataka potrebno je instalirati Npgsql dodatak koji onda omogućava funkcije za rad s PostgreSQL bazama podataka.



Slika 5: Visual Studio (Vlastita izrada)

4.2. Izrada baze podataka

U ovom poglavlju je opisan svaki atribut svake tablice koje se nalaze u bazi podataka te je također uz to priložen i dio SQL koda potreban za kreiranje navedene tablice. SQL kod je kreiran pomoću alata Navicat 15. U kodu se mogu vidjeti određena dodatna ograničenja i svojstva atributa koja nije moguće prikazati u ERA modelu.

• Komponenta

- **ID komponente** - Jedinstvena oznaka komponente.
- **Naziv** - Naziv komponente.
- **Cijena** - Cijena komponente.
- **Kolicina Narudzbe** - Količina koja će se naručiti nakon što se stanje smanji ispod minimalne potrebne količine na skladištu koja se nalazi u atributu "Min kolicina".
- **Min kolicina** - Minimalna količina koja uvijek mora biti na skladištu. Ako ima manje, onda se mora naručiti količina komponenti u visini atributa "Kolicina Narudzbe".

- **VrstaID** - Oznaka vrste komponente, vanjski ključ na tablicu "Vrsta Komponente".
- **ProizvodacID** - Oznaka proizvođača, vanjski ključ na tablicu "Proizvodac Komponente".
- **KarakteristikaID** - Oznaka karakteristike komponente, vanjski ključ na tablicu "Karakteristika Komponente".

```
CREATE TABLE "public"."Komponenta" (
  "ID_komponente" int4 NOT NULL DEFAULT nextval('id_komp_auto_ink'::regclass)
  ,
  "Naziv" varchar(255) COLLATE "pg_catalog"."default" NOT NULL,
  "Cijena" money,
  "Kolicina_Narudzbe" int2,
  "Min_kolicina" int2 NOT NULL,
  "VrstaID" int4,
  "ProizvodacID" int4,
  "KarakterisikeID" int4,
  CONSTRAINT "Komponenta_pkey" PRIMARY KEY ("ID_komponente"),
  CONSTRAINT "KarakteristikeID" FOREIGN KEY ("KarakterisikeID") REFERENCES "
    public"."Karakteristike_Komponente" ("ID_karakteristika") ON DELETE NO
    ACTION ON UPDATE NO ACTION,
  CONSTRAINT "ProizvodacID" FOREIGN KEY ("ProizvodacID") REFERENCES "public"."
    Proizvodac_Komponente" ("ID_proizvodaca") ON DELETE NO ACTION ON
    UPDATE NO ACTION,
  CONSTRAINT "VrstaID" FOREIGN KEY ("VrstaID") REFERENCES "public"."
    Vrsta_Komponente" ("ID_vrste") ON DELETE NO ACTION ON UPDATE NO ACTION
)
;
```

• Vrsta Komponente

- **ID vrste** - Jedinstvena oznaka vrste.
- **Naziv** - Naziv vrste.
- **Opis** - Opis vrste.

```
CREATE TABLE "public"."Vrsta_Komponente" (
  "ID_vrste" int4 NOT NULL DEFAULT nextval('id_vrsta_auto_ink'::regclass),
  "Naziv" varchar(255) COLLATE "pg_catalog"."default",
  "Opis" varchar(255) COLLATE "pg_catalog"."default",
  CONSTRAINT "Vrsta_Komponente_pkey" PRIMARY KEY ("ID_vrste"),
  CONSTRAINT "Jedinstven" UNIQUE ("Naziv")
)
;
```

• Stanje Skladista

- **KomponentalID** - Primarni ključ je ujedno i vanjski ključ na tablicu "Komponenta" jer u ovoj tablici ne smije postojati stavka koja nije u tablici "Komponenta".
- **Kolicina** - Trenutna količina računalne komponente na skladištu.

```
CREATE TABLE "public"."Stanje_Skladista" (
  "KomponentaID" int4 NOT NULL,
  "Kolicina" int2,
  CONSTRAINT "Stanje_Skladista_pkey" PRIMARY KEY ("KomponentaID"),
  CONSTRAINT "KomponentaID" FOREIGN KEY ("KomponentaID") REFERENCES "public".
    "Komponenta" ("ID_komponente") ON DELETE NO ACTION ON UPDATE NO ACTION
)
;
```

• Proizvodac Komponente

- **ID proizvodaca** - Jedinstvena oznaka proizvođača.
- **Naziv** - Naziv proizvođača računalne komponente.

```
CREATE TABLE "public"."Proizvodac_Komponente" (
  "ID_proizvodaca" int4 NOT NULL DEFAULT nextval('id_proizv_auto_ink'::
    regclass),
  "Naziv" varchar(255) COLLATE "pg_catalog"."default",
  CONSTRAINT "Proizvodac_Komponente_pkey" PRIMARY KEY ("ID_proizvodaca")
)
;
```

• Nabava Komponenti

- **ID nabave** - Jedinstvena oznaka nabave.
- **Trenutno stanje** - Kolicina na skladištu u trenutku kada treba naručiti komponentu.
- **Datum trenutnog stanja** - Datum kada se stanje u skladištu smanji ispod minimalne količine.
- **Kolicina za nabaviti** - Količina pojedine komponente koja se mora naručiti.
- **Datum Zaprimanja** - Datum kada je narudžba stigla od proizvođača.
- **KomponentaID** - Oznaka komponente koju treba naručiti, vanjski ključ na tablicu "Komponenta".
- **Dostavljeno** - Potvrda da je narudžba stigla.

```
CREATE TABLE "public"."Nabava_Komponenti" (
  "ID_nabave" int4 NOT NULL DEFAULT nextval('id_nabava_auto_ink'::regclass),
  "Trenutno_stanje" int4 NOT NULL,
  "Datum_trenutnog_stanja" timestamp(6) NOT NULL DEFAULT now(),
  "Kolicina_za_nabaviti" int2,
  "Datum_Zaprimanja" date,
  "KomponentaID" int4 NOT NULL,
  "Dostavljeno" varchar(255) COLLATE "pg_catalog"."default",
  CONSTRAINT "Nabava_Komponenti_pkey" PRIMARY KEY ("ID_nabave"),
  CONSTRAINT "KomponentaID" FOREIGN KEY ("KomponentaID") REFERENCES "public".
    "Komponenta" ("ID_komponente") ON DELETE NO ACTION ON UPDATE NO ACTION
)
;
```

• Karakteristike Komponente

- **ID karakteristike** - Jedinstvena oznaka karakteristike.
- **God proizvodnje** - Godina proizvodnje pojedine komponente.
- **TDP** - Potrošnja el. energije pojedine komponente.
- **Garancija** - Duljina garancije pojedine komponente.
- **ProizvodniProces** - Proizvodni proces pojedine komponente

```
CREATE TABLE "public"."Karakteristike_Komponente" (  
  "ID_karakteristika" int4 NOT NULL DEFAULT nextval('id_karak_auto_ink'::  
    regclass),  
  "God_proizvodnje" date,  
  "TDP" varchar(255) COLLATE "pg_catalog"."default",  
  "Garancija" varchar(255) COLLATE "pg_catalog"."default",  
  "ProizvodniProces" varchar(255) COLLATE "pg_catalog"."default",  
  CONSTRAINT "Karakteristike_Komponente_pkey" PRIMARY KEY ("ID_karakteristika"  
    )  
)  
;
```

• Evidencija Skladista

- **ID evidencije** - Jedinstvena oznaka evidencije.
- **Datum** - Datum kada se promjena dogodila.
- **KolicinaPrije** - Količina komponente na skladištu prije promjene.
- **KolicinaNova** - Količina komponente na skladištu nakon promjene.
- **KomponentaID** - Oznaka komponente kojoj se stanje mijenja, vanjski ključ na tablicu "Komponenta".

```
CREATE TABLE "public"."Evidencija_Skladista" (  
  "ID_evidencije" int4 NOT NULL DEFAULT nextval('id_evid_auto_ink'::regclass)  
    , "Datum" timestamp(6) NOT NULL DEFAULT now(),  
  "KolicinaPrije" int4 DEFAULT 0,  
  "KolicinaNova" int4 NOT NULL,  
  "KomponentaID" int4 NOT NULL,  
  CONSTRAINT "Evidencija_Skladista_pkey" PRIMARY KEY ("ID_evidencije"),  
  CONSTRAINT "KomponentaID" FOREIGN KEY ("KomponentaID") REFERENCES "public"  
    "."Komponenta" ("ID_komponente") ON DELETE NO ACTION ON UPDATE NO  
    ACTION  
)  
;
```

• Narudžbenica

- **ID narudžbenice** - Jedinstvena oznaka narudžbenice.
- **Datum** - Datum kada je narudžbenica napravljena.
- **Kolicina** - Količina komponente koja je naručena.

- **NabavaID** - Oznaka nabave koja je kreirala narudžbenicu, vanjski ključ na tablicu "Nabava Komponenti".
- **KomponentaID** - Oznaka komponente koja je naručena, vanjski ključ na tablicu "Komponenta".

```
CREATE TABLE "public"."Narudzbenica" (
  "ID_narudzbenice" int4 NOT NULL DEFAULT nextval('id_narudzb_auto_ink'::
    regclass),
  "Datum" timestamp(6) DEFAULT now(),
  "Kolicina" varchar(255) COLLATE "pg_catalog"."default",
  "NabavaID" int4,
  "KomponentaID" int4,
  CONSTRAINT "Narudzbenica_pkey" PRIMARY KEY ("ID_narudzbenice"),
  CONSTRAINT "KomponentaID" FOREIGN KEY ("KomponentaID") REFERENCES "public".
    "Komponenta" ("ID_komponente") ON DELETE NO ACTION ON UPDATE NO ACTION,
  CONSTRAINT "NabavaID" FOREIGN KEY ("NabavaID") REFERENCES "public"."
    Nabava_Komponenti" ("ID_nabave") ON DELETE NO ACTION ON UPDATE NO
    ACTION
)
;
```

4.3. Okidači

Kreiranje okidača nad određenim tablicama pomaže u automatizaciji cijelog skladišta. Korisniku je olakšan rad s aplikacijom, a paralelno je smanjena mogućnost pogreške tijekom rada skladišta. U sljedećim primjerima okidača i okidačkih funkcija prikazat će se koji su točno procesi automatizirani, a popratni kod okidača i pripadne okidačke funkcije će pokazati na koji način je to točno postignuto. U nekoliko slučajeva se povezuju aktivna i temporalna komponenta baze podataka.

4.3.1. Evidencija ažuriranja količine komponenti

Kada korisnik unese novo stanje količine za neku računalnu komponentu koja je već u skladištu onda se automatski pokreće okidač i njegova funkcija koja evidentira promjenu u tablici "Evidencija Skladišta".

Okidač

```
CREATE TRIGGER "evidentiraj" AFTER UPDATE ON "public"."Stanje_Skladista"
FOR EACH ROW
EXECUTE PROCEDURE "public"."evidentiraj_promjenu"();
```

Okidačka funkcija

```
CREATE OR REPLACE FUNCTION "public"."evidentiraj_promjenu"()
RETURNS "pg_catalog"."trigger" AS $BODY$
BEGIN
```

```

INSERT INTO "Evidencija_Skladista"("KolicinaPrije", "KolicinaNova", "
KomponentaID")
VALUES(OLD."Kolicina", NEW."Kolicina", OLD."KomponentaID");
RETURN NEW;
END$BODY$
LANGUAGE plpgsql VOLATILE
COST 100;

```

4.3.2. Evidencija brisanja komponente sa skladišta

Prilikom brisanja neke komponente sa skladišta, u slučaju kada se ta komponenta više neće držati u skladištu, paralelno se aktivira okidač i ta promjena se evidentira u tablicu "Evidencija Skladišta".

Okidač

```

CREATE TRIGGER "evidentiraj_brisanje_komp" AFTER DELETE ON "public"."
Stanje_Skladista"
FOR EACH ROW
EXECUTE PROCEDURE "public"."evidentiraj_brisanje"();

```

Okidačka funkcija

```

CREATE OR REPLACE FUNCTION "public"."evidentiraj_brisanje"()
RETURNS "pg_catalog"."trigger" AS $BODY$
BEGIN
INSERT INTO "Evidencija_Skladista"("KolicinaPrije", "KolicinaNova", "
KomponentaID")
VALUES(OLD."Kolicina", 0, OLD."KomponentaID");
RETURN NEW;
END$BODY$
LANGUAGE plpgsql VOLATILE
COST 100;

```

4.3.3. Evidencija nove komponente na skladište

Kada se u skladište dopremi komponenta koja nije do sada postojala u skladištu, korisnik ju unosi u skladište, a uz to se pokreće okidač koji to evidentira u tablici "Evidencija skladišta".

Okidač

```

CREATE TRIGGER "evidentiraj_nova_komp" AFTER INSERT ON "public"."
Stanje_Skladista"
FOR EACH ROW
EXECUTE PROCEDURE "public"."evidentiraj_promjenu_nova_komp"();

```

Okidačka funkcija

```

CREATE OR REPLACE FUNCTION "public"."evidentiraj_promjenu_nova_komp"()
RETURNS "pg_catalog"."trigger" AS $BODY$

```

```

BEGIN
    INSERT INTO "Evidencija_Skladista" ("KolicinaNova", "KomponentaID")
    VALUES (NEW."Kolicina", NEW."KomponentaID");
    RETURN NEW;
END$BODY$
LANGUAGE plpgsql VOLATILE
COST 100;

```

4.3.4. Stvaranje narudžbe

Kada korisnik ažurira stanje na skladištu i dogodi se da nova količina komponente je manja od minimalne količine koja uvijek mora biti na skladištu, onda se automatski pokreće okidač koji stvara stavku u tablici "Nabava Komponenti". Ta stavka u tablici znači da je to jedna od komponenti koja se mora naručiti no važno je istaknuti da to ne znači da je i automatski istog trenutka naručena. S ovim okidačem se postiže instantno kreiranje narudžbe za određenu komponentu te se ne ovisi o korisniku da prati stanje minimalnih količina i trenutnih količina.

Okidač

```

CREATE TRIGGER "nabavi_komp" AFTER UPDATE ON "public"."Stanje_Skladista"
FOR EACH ROW
EXECUTE PROCEDURE "public"."za_nabaviti"();

```

Okidačka funkcija

```

CREATE OR REPLACE FUNCTION "public"."za_nabaviti"()
RETURNS "pg_catalog"."trigger" AS $BODY$
    DECLARE kolZaNabav INTEGER;
    DECLARE minKol INTEGER;
BEGIN
    SELECT "Komponenta"."Kolicina_Narudzbe" INTO kolZaNabav FROM "Komponenta"
    WHERE "ID_komponente" = OLD."KomponentaID";
    SELECT "Komponenta"."Min_kolicina" INTO minKol FROM "Komponenta" WHERE "
    ID_komponente" = OLD."KomponentaID";
    IF (NEW."Kolicina" < minKol) THEN
        INSERT INTO "Nabava_Komponenti" ("Trenutno_stanje", "
        Kolicina_za_nabaviti", "KomponentaID", "Dostavljeno")
        VALUES (NEW."Kolicina", kolZaNabav, OLD."KomponentaID", 'false');
    END IF;
    RETURN NEW;
END$BODY$
LANGUAGE plpgsql VOLATILE
COST 100;

```

4.3.5. Ažuriranje stanja skladišta sa dostavljenom komponentom

Kada korisnik potvrdi da je izvršena narudžbenica i da je komponenta stigla u skladište, pokreće se okidač koji ažurira dostavljenu količinu na skladištu u tablicu "Stanje skladista". Pomoću ove funkcije sprječava se mogućnost pogreške koja uvijek postoji kada korisnik sam unosi podatke.

Okidač

```
CREATE TRIGGER "zavrshi_dostavu_u_sk" AFTER DELETE ON "public"."Narudzbenica"
FOR EACH ROW
EXECUTE PROCEDURE "public"."unesi_u_skladiste"();
```

Okidačka funkcija

```
CREATE OR REPLACE FUNCTION "public"."unesi_u_skladiste"()
RETURNS "pg_catalog"."trigger" AS $BODY$
BEGIN
    UPDATE "Stanje_Skladista"
    SET "Kolicina" = "Kolicina" + CAST(OLD."Kolicina" AS INTEGER)
    WHERE "KomponentaID" = OLD."KomponentaID";
    RETURN NEW;
END$BODY$
LANGUAGE plpgsql VOLATILE
COST 100;
```

4.3.6. Evidentiranje izvršene narudžbe

Kada korisnik potvrdi da je izvršena narudžbenica prilikom čega se briše narudžbenica iz tablice "Narudzbenica", pokreće se okidač koji ažurira stanje narudžbe u tablici "Nabava Komponenti". Ovim se sprječava da u redu za naručivanje komponenti ostane stavka koja je već dostavljena i izvršena.

Okidač

```
CREATE TRIGGER "zavrshi_nabavu" AFTER DELETE ON "public"."Narudzbenica"
FOR EACH ROW
EXECUTE PROCEDURE "public"."zabiljezi_dostavu"();
```

Okidačka funkcija

```
CREATE OR REPLACE FUNCTION "public"."zabiljezi_dostavu"()
RETURNS "pg_catalog"."trigger" AS $BODY$
BEGIN
    UPDATE "Nabava_Komponenti"
    SET "Dostavljeno" = 'true'
    WHERE "ID_nabave" = OLD."NabavaID";
    RETURN NEW;
END$BODY$
LANGUAGE plpgsql VOLATILE
COST 100;
```

4.4. Aplikacija

U ovom poglavlju prikazat će se isječci koda koji su bitni za osnovne interakcije aplikacije i baze podataka kao što su spajanje, unos, brisanje i uređivanje podataka.

4.4.1. Spajanje sa bazom podataka

Jedan od najvažnijih dijelova koda za funkcioniranje cijelog projekta je ova poveznica između baze podataka i aplikacija. Dodatak koji je spomenut ranije, Npgsql, je potreban za upravljanje konekcijom. Ovdje su kreirane dvije funkcije koje služe otvaranje i zatvaranje veze na bazu. Također ovdje se upisuju ključni podaci koje smo prije definirali u pgAdmin4 alatu kao što su lokacija servera, port koji se koristi, lozinka i baza koja se namjerava koristiti.

```
public class SpajanjeNaDB
{
    public NpgsqlConnection connection;

    public void SpojiNaBazu()
    {
        string ConnectionString = "Server=localhost;Port=5432;User_Id=postgres;
        Password_=lozinka2;Database_=šSkladiteKomponenti;";
        connection = new NpgsqlConnection(ConnectionString);
        connection.Open();
    }

    public void OdspojiSBazu()
    {
        connection.Close();
    }
}
```

4.4.2. Dohvaćanje podataka iz baze

Da bi dohvatili podatke iz baze podataka moraju se kombinirati kod aplikacije i kod baze podataka. Specifično, varijabla "sql" je zaslužna za upit koji će se napraviti kada se uspostavi konekcija aplikacije i baze. Tada rezultati upita koji će biti u ovom slučaju podaci iz tablice "Komponenta" pretvaraju u oblik koji aplikacija može koristiti i popunjava se objekt "dgvKomponente" koji će onda u formi aplikacije prikazivati dohvaćene podatke.

```
private void UcitajSkladiste(NpgsqlConnection connection)
{
    string sql = "SELECT \"Komponenta\".\"ID_komponente\", \"Komponenta\".\"Naziv\", \"Komponenta\".\"Min_kolicina\", \"Stanje_Skladista\".\"Kolicina\" +
        \"FROM \"Komponenta\" +
        \"INNER_JOIN \"Stanje_Skladista\" +
        \"ON \"Komponenta\".\"ID_komponente\"= \"Stanje_Skladista\".\"KomponentaID\" +
        \"ORDER_BY \"Komponenta\".\"ID_komponente\" ASC";

    NpgsqlDataAdapter dataAdapter = new NpgsqlDataAdapter(sql, connection);
    DataSet podaciKomponentata = new DataSet();

    dataAdapter.Fill(podaciKomponentata);
    dgvKomponente.DataSource = podaciKomponentata.Tables[0];
}
```

4.4.3. Unos podataka u bazu

Unos podataka se odvija slično kao i dohvaćanje no važno je to što se naredba mijenja u varijabli "sql" te je također potrebno uvrstiti lokalne varijable koje postoje u aplikaciji da bi se unijeli novi podaci kreirani u aplikaciji, u bazu podataka.

```
spajanjeNaDB.SpojNaBazu();

string sql = "INSERT INTO \"Stanje_Skladista\"_\" +
            \"VALUES_\" + oznakaKomponente + \",\" + novaKolicina + \");";

NpgsqlCommand command = new NpgsqlCommand(sql, spajanjeNaDB.connection);
try
{
    command.ExecuteNonQuery();
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message, "Something_happened!", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
}

UcitajSkladiste(spajanjeNaDB.connection);
UcitajEvidenciju(spajanjeNaDB.connection);
spajanjeNaDB.OdspojSBazu();
```

4.4.4. Ažuriranje baze podataka

Ažuriranje podataka koristi novu naredbu u varijabli "sql" te je također važno da se pomoću funkcije "UcitajSkladiste" ponovno učitaju podaci da bi se prikazala promjena.

```
spajanjeNaDB.SpojNaBazu();

string sql = "UPDATE_\"Stanje_Skladista\"_\" +
            \"SET_\"Kolicina_\"_\" + novaKolicina +
            \"WHERE_\"KomponentaID_\"_\" + oznakaKomponente + \");";

NpgsqlCommand command = new NpgsqlCommand(sql, spajanjeNaDB.connection);

try
{
    command.ExecuteNonQuery();
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message, "Something_happened", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
}

UcitajSkladiste(spajanjeNaDB.connection);
UcitajEvidenciju(spajanjeNaDB.connection);
spajanjeNaDB.OdspojSBazu();
```

4.4.5. Brisanje podataka u bazi

Kod brisanja podataka bitno je dobro navesti lokalne varijable koje će se koristiti u varijabli "sql" jer korištenjem krivih rezultira pogrešnim brisanjem u bazi podataka.

```
spajanjeNaDB.SpojNaBazu();

string sql = "DELETE_" +
            "FROM_" + Stanje_Skladista "_" +
            "WHERE_" + Stanje_Skladista "." + KomponentaID "_" + oznakaKomponente
            + "_";

NpgsqlCommand command = new NpgsqlCommand(sql, spajanjeNaDB.connection);

try
{
    command.ExecuteNonQuery();
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message, "Something_happened", MessageBoxButtons.OK,
                    MessageBoxIcon.Error);
}

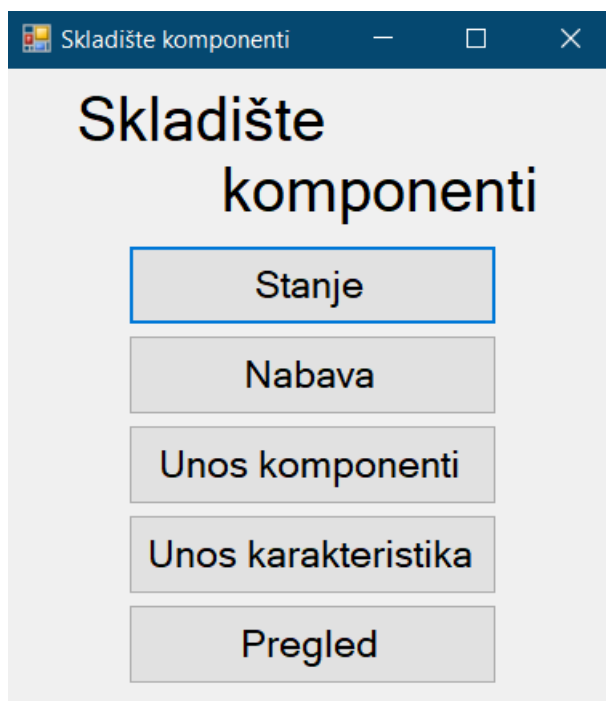
UcitajSkladiste(spajanjeNaDB.connection);
UcitajEvidenciju(spajanjeNaDB.connection);
spajanjeNaDB.OdspojisBazu();
```

5. Primjeri korištenja

Ovo poglavlje služi kao primjer korištenja aplikacije i njezinih mogućnosti. Objasnit će se funkcije koje svaka forma ima te kako se otprilike treba odvijati korištenje tih funkcija.

5.1. Glavni izbornik

Prilikom otvaranja aplikacije susrećemo se s glavnim izbornikom aplikacije. Glavni izbornik aplikacije je napravljen da bude što jednostavniji i što razumljiviji za krajnjeg korisnika. Sadrži gumbove koji kada se pritisnu vode na ostale forme za najvažnije funkcije skladišta i ništa više se ne nalazi na formi jer ova aplikacija treba prije svega biti funkcionalna, a manje se uzima u obzir izgled.



Slika 6: Glavni izbornik (Vlastita izrada)

5.2. Stanje skladišta

Kada je korisnik iz glavnog izbornika odabrao opciju "Stanje", otvara se nova forma "Stanje skladišta" koja prikazuje najvažnije informacije o trenutnom stanju skladišta. Prva funkcionalnost je dodavanje nove komponente na skladište. Kako bi se to napravilo mora se u okvir "Oznaka komponente" unijeti potreban broj te u okvir "Nova količina" je potrebno unijeti količinu komponente koja će se nalaziti na skladištu. Nakon toga mora se kliknuti gumb "Dodaj" i komponenta će biti unesena. Kada se želi ažurirati postojeću komponentu mora se prvo označiti komponenta u tablici "Trenutno na skladištu" i unijeti u okvir "Nova količina" novu količinu komponente. Nakon toga se klikne gumb "Ažuriraj" i to je to. U slučaju da se neka komponenta više

neće držati na skladištu onda se pomoću odabira u tablici "Trenutno na skladištu" i pritiskom na gumb "Briši" briše komponenta sa skladišta. Sve ove promjene se automatski bilježe u tablici "Promjene stanja skladišta".

Sljedeća mogućnost je pregled svih promjena stanja komponenata u skladištu. To se može napraviti pregledom tablice "Promjene stanja skladišta". Ako se traži neki poseban datum i sve promjene na taj datum ili interval između neka dva datuma onda je potrebno namjestiti u "Search" odjeljku "Od" i "Do" datume. Prilikom podešavanja tih datuma automatski se u tablici prikazuju tražene stavke. Ako nakon tog pregleda želimo opet pogledati sve promjene potrebno je samo kliknuti na gumb "Prikaži sve promjene" i tablica će pokazivati najnovije stanje.

Stanje skladišta

Ažuriranje količine:

Oznaka komponente:

Nova količina:

Dodaj

Ažuriraj

Briši

Trenutno na skladištu:

	Oznaka	Naziv	Minimalna potrebna količina	Trenutna količina
▶	1	Core i7 10900k	2	10
	2	Core i5 10400	5	8
	3	Core i3 10100	10	27
	4	Pentium G6400	15	40
	5	Ryzen 9 5900X	2	4
	6	Ryzen 7 5800X	2	5
	7	Ryzen 5 5600X	5	14
	9	RTX 3080	1	5
	10	RTX 3070	2	6

Search:

Od:

Do:

Prikaži sve promjene

Promjene stanja skladišta:

	Datum	Stara količina	Nova količina	Naziv
▶	23.1.2021. 21:17	4	14	Ryzen 5 5600X
	23.1.2021. 21:16	4	4	Ryzen 5 5600X
	23.1.2021. 21:15	6	0	RX 6800XT
	23.1.2021. 21:15	5	6	RX 6800XT
	23.1.2021. 21:15	0	5	RX 6800XT
	20.1.2021. 17:25	5	0	RTX 3090
	20.1.2021. 17:25	0	10	RX 6900XT
	20.1.2021. 17:25	20	10	Core i7 10900k
	20.1.2021. 15:17	10	40	Pentium G6400
	20.1.2021. 15:16	10	10	Pentium G6400

Slika 7: Stanje skladišta (Vlastita izrada)

5.3. Nabava komponenti

Forma "Nabava komponenti" služi za olakšavanje naručivanja potrebnih komponenata. Kada se u formi "Stanje skladišta" napravi promjena koja rezultira količinom neke komponente ispod njezine minimalne količine na skladištu, automatski se kreira stavka u formi "Nabava

komponenti", točnije u tablici "Za nabaviti". Kada korisnik želi naručiti potrebne komponente dovoljno je samo označiti stavku u tablici "Za nabaviti" i pritisnuti gumb "Naruči". Nakon toga se kreira narudžbenica u tablici "Narudžbenice u tijeku" koja predstavlja da je neka komponenta naručena i da se čeka da se dopremi. Sljedeća funkcionalnost ove forme je potvrđivanje da je neka pošiljka došla. U tablici "Narudžbenice u tijeku" potrebno je označiti stavku koja je došla i pritisnuti gumb "Dostavljeno". Prilikom pritiska se događa nekoliko akcija. Prva akcije je brisanje narudžbenice i automatsko ažuriranje stanja u formi "Stanje skladišta", točnije tablici "Trenutno na skladištu" i automatski se dogodi da se pokrene i ažuriranje tablice "Promjene stanja skladišta" u istoj formi. Druga akcija je da se stavka koja se nalazi u tablici "Za nabaviti" ažurira da je dostavljena pošiljka i ona prebacuje u tablicu "Povijest narudžbi".

Za nabaviti:

	Oznaka	Trenutno stanje	Datum trenutnog stanja	Kolicina za nabaviti	Datum zaprimanja	Oznaka komponente	Dostavljeno
»							

Naruči

Narudžbenice u tijeku:

	Oznaka	Datum	Kolicina	Oznaka nabave	Oznaka komponente
*					

Dostavljeno

Povijest narudžbi:

	Oznaka	Trenutno stanje	Datum trenutnog stanja	Kolicina za nabaviti	Datum zaprimanja	Oznaka komponente	Dostavljeno
▶	1	10	17.1.2021. 13:34	30	20.1.2021.	4	true
	3	10	19.1.2021. 19:51	12		4	true
	4	10	20.1.2021. 12:30	15		3	true
	6	10	20.1.2021. 12:44	10		1	true
	7	7	20.1.2021. 15:12	20		2	true

Slika 8: Nabava komponenti (Vlastita izrada)

5.4. Unos komponenti

Pošto skladište računalnih komponenti stalno dobiva neke nove komponente na skladište potrebno je nekako i zapisati tu novu komponentu. Korisnik kada otvori formu "Unos komponenti" može napraviti sljedeće. Da bi se olakšao unos napravljene su tablice s dostupnim podacima kao što su "Vrste", "Proizvođači", "Karakteristike". Korisnik prvo treba upisati "Naziv", "Cijenu", "Količinu narudžbe" i "Minimalnu količinu", a onda odabrati iz dostupnih tablica ostale podatke. Nakon što je sve uneseno i odabrano treba se pritisnuti gumb "Unesi".

Unos komponenti

Podaci:

Naziv:

Cijena:

Količina narudžbe:

Minimalna količina:

Vrsta:

Unesi

Vrste:

	Oznaka	Naziv	Opis
▶	1	CPU	Procesor je mozak računala koji izvršava sve aktivnosti koje
	2	GPU	Graficka kartica služi za prikaz slike na monitoru
	3	RAM	RAM je memorija koja u sebi drži informacije trenutno potrebr
	4	PSU	Mrežna računala služi za povezivanje različitih komponenta sa

Proizvođači:

	Oznaka	Naziv
▶	1	Intel
	2	AMD
	3	NVIDIA
	4	Kingston

Karakteristike:

	Oznaka	Godina proizvodnje	TDP	Garancija	Proizvodni proces
▶	1	10.12.2020.	125	5	14nm
	2	10.11.2020.	65	5	14nm
	3	18.6.2020.	105	5	7nm
	4	17.12.2020.	65	5	7nm

Slika 9: Unos komponenti (Vlastita izrada)

5.5. Unos karakteristika

Unos karakteristika je jedna od formi koja dopušta dodatno specificiranje proizvoda koji do sada nije bio na skladištu. U tablici "Karakteristike" se mogu vidjeti karakteristike do sada unesenih komponenata na skladište. Za unos novih karakteristika je potrebno unijeti u okvire "Godina proizvodnje", "TDP", "Garancija", "Proizvodni proces" podatke i pritisnuti gumb "Unesi".

Unos karakteristika

Podaci:

Godina proizvodnje:

TDP:

Garancija:

Proizvodni proces:

Unesi

Karakteristike:

	Oznaka	Godina proizvodnje	TDP	Garancija	Proizvodni proces
▶	1	10.12.2020.	125	5	14nm
	2	10.11.2020.	65	5	14nm
	3	18.6.2020.	105	5	7nm
	4	17.12.2020.	65	5	7nm
	5	30.12.2020.	35	5	10nm
	6	17.11.2020.	28	5	10nm
	7	21.10.2020.	350	5	8nm
	8	4.11.2020.	300	5	8nm
	9	17.11.2020.	220	5	8nm
	10	14.9.2020.	15	5	10nm
	11	6.1.2021.	1000	10	
	12	13.1.2021.	750	10	
	13	6.1.2021.	650	10	
	14	29.12.2020.	500	10	

Slika 10: Unos karakteristika (Vlastita izrada)

5.6. Pregled svih komponenti

Forma "Pregled svih komponenti" dopušta pretraživanje svih komponenti koje su do sada bile u skladištu. Postoje razne opcije po kojima se može pretraživati. Ono što je potrebno je odabrati željenu stavku po kojoj se želi pretražiti i odabrati ponuđena vrijednost koja se izlista prilikom pritiska. Kada se napravi odabir tablica će automatski pokazati sve komponente s tim obilježjem. Kada je korisnik želi ponovno vidjeti cijeli popis to može učiniti pritiskom na gumb "Prikaži sve".

Pregled komponenti

Search:

Proizvođač:

Naziv:

Vrsta:

TDP:

Proizvodni proces:

Prikaži sve

Pregled svih komponenti:

	Oznaka	Proizvođač	Naziv	Vrsta	Cijena	Godina proizvodnje	TDP	Garancija	Proizvodni proces
▶	1	Intel	Core i7 10900k	CPU	5000,00	10.12.2020.	125	5	14nm
	2	Intel	Core i5 10400	CPU	3000,00	10.11.2020.	65	5	14nm
	3	Intel	Core i3 10100	CPU	1000,00	10.11.2020.	65	5	14nm
	4	Intel	Pentium G6400	CPU	500,00	10.11.2020.	65	5	14nm
	5	AMD	Ryzen 9 5900X	CPU	5500,00	18.6.2020.	105	5	7nm
	6	AMD	Ryzen 7 5800X	CPU	4500,00	18.6.2020.	105	5	7nm
	7	AMD	Ryzen 5 5600X	CPU	3500,00	18.6.2020.	105	5	7nm
	8	NVIDIA	RTX 3090	GPU	10000,00	21.10.2020.	350	5	8nm
	9	NVIDIA	RTX 3080	GPU	8000,00	4.11.2020.	300	5	8nm
	10	NVIDIA	RTX 3070	GPU	5000,00	17.11.2020.	220	5	8nm
	11	AMD	RX 6900XT	GPU	10000,00	4.11.2020.	300	5	8nm
	12	AMD	RX 6800XT	GPU	7000,00	4.11.2020.	300	5	8nm
	13	AMD	RX 5700XT	GPU	4000,00	17.11.2020.	220	5	8nm
	14	Intel	Core i5 10600k	CPU	3500,00	10.12.2020.	125	5	14nm
	15	AMD	FX 9590	CPU	500,00	20.1.2021.	250	5	28
	16	AMD	FX 8320	CPU	200,00	20.1.2021.	250	5	28
*									

Slika 11: Pregled komponenti (Vlastita izrada)

22

6. Zaključak

Cilj projekta je bio kreiranje aplikacije i baze podataka za upravljanje skladištem. Ono što se ovom aplikacijom postiglo je jednostavnije upravljanje skladištem i određena doza automatizacije korištenjem aktivnih baza podataka. Korisnik s ovom aplikacijom može jednostavno unositi, ažurirati, brisati komponente, a uz to i olakšano nabavljati potrebne komponente. Također, u projektu je olakšana i analiza skladišta korištenjem temporalnih baza podataka tako što se pruža uvid u prijašnje stanje skladišta. Aplikacija je dizajnirana da bude lako razumljiva za korištenje tako što su sve tablice i polja jasno objašnjena.

Tijekom izrade projekta korišteni su neki novi alati koji su rezultirali lakšom i kvalitetnijom izradom projekta. U konačnici, projekt je bio zanimljiv zbog potrebe korištenja naprednijih tehnologija, a i zbog realne primjene u stvarnom svijetu.

Popis literature

- [1] B. O. Đurić, *Aktivne Baze Podataka*, <https://files.classcraft.com/game/uploads/zijp9vaaGwK86piNu/1571991887636/tbp2019-003-ABP.pdf>, Accessed on 2021-20-01.
- [2] —, *Temporalne Baze Podataka*, <https://files.classcraft.com/game/uploads/zijp9vaaGwK86piNu/1572000575184/tbp2019-005-TempBP.pdf>, Accessed on 2021-20-01.
- [3] pgAdmin, *pgAdmin - PostgreSQL Tools*, <https://www.pgadmin.org/>, Accessed on 2021-19-01.
- [4] Navicat, *Navicat*, <https://www.navicat.com/en/navicat-15-highlights>, Accessed on 2021-19-01.
- [5] Microsoft, *Visual Studio IDE*, <https://visualstudio.microsoft.com/>, Accessed on 2021-19-01.

Popis slika

1.	ERA model (Vlastita izrada)	3
2.	Postavke veze (Vlastita izrada)	5
3.	Informacije o bazi (Vlastita izrada)	6
4.	Prikaz tablica u alatu Navicat 15 (Vlastita izrada)	6
5.	Visual Studio (Vlastita izrada)	7
6.	Glavni izbornik (Vlastita izrada)	18
7.	Stanje skladišta (Vlastita izrada)	19
8.	Nabava komponenti (Vlastita izrada)	20
9.	Unos komponenti (Vlastita izrada)	21
10.	Unos karakteristika (Vlastita izrada)	21
11.	Pregled komponenti (Vlastita izrada)	22

Popis tablica

1. Prilog 1

2. Prilog 2