Objektovo-orientované programovanie

Riadiaci softvér pre lokálnu handmade výrobu svetrov

Emma Macháčová

Meno cvičiaceho : Ing. Anna Považanová
Čas cvičení : štvrtok 9:00
Dátum vytvorenia : 03. marec 2021

Obsah

Zámer projektu

Softvér má slúžiť na manažovanie procesov, ktoré sú spojené s výrobou a následným predajom handmade svetrov lokálnou spoločnosťou.

Medzi tieto procesy patrí najmä správa inventáru spoločnosti, tvorba interných objednávok spojených so zabezpečovaním nákupu materiálu potrebného na výrobu, samotná výroba konečných produktov, generovanie časového harmonogramu výroby, rozdeľovanie úloh medzi zamestnancov a konečné odoslanie výrobku zákazníkovi. Tak isto je cieľom správa toku informácií medzi jednotlivými oddeleniami a kľúčovými zamestnancami. Zamestnanci pracujú na rôznych typoch pracovných pozícií, ich schopnosti a úlohy sa líšia a sú platení na základe ich pracovných výkonov. Výsledným produktom výrobného procesu je pletený výrobok, predovšetkým kus odevu (sveter), ale je možné vyrábať aj iné produkty (vesty, tašky, čiapky, rukavice..).

Úlohou softvér je priblížiť sa k optimálnej výrobe (všetky zakúpené materiály použité a všetky vyrobené svetre predané). Systém nebude brať do úvahy čas potrebný na dodanie objednaného materiálu (tovar je k dispozícii ihneď po objednaní), ani na čas potrebný na konečný export produktov zákazníkovi (sklad je uvoľnený momentom predaja). Koncom každého dňa systém vyhodnotí efektivitu výroby. Softvér nebude brať do úvahy prípadné meškanie zamestnanca alebo PN (všetci zamestnanci pracujú neustále).

Kľúčové slová

- Sveter – výsledný produkt výrobného procesu, má určitú veľkosť, strih, a iné vlastnosti závisiace od materiálu z ktorého je vytvorený
- Klbko – materiál potrebný pre vytvorenie výsledného produktu, má vlastnosti ako farbu, zloženie, gramáž, priemernú spotrebu, cenu za kus
- Sklad - priestor na uloženie tovaru, surovín, materiálu
- Zamestnanec – fyzická osoba, ktorá v konkrétnom pracovnoprávnom vzťahu vykonáva pre zamestnávateľa závislú prácu, určenú v závislosti od typu zamestnanca
- Výrobok – predmet kúpy a predaja (svetre, tašky, čiapky..)
- Vybavenie – nástroje potrebné na výrobný proces (rovné ihlice, kruhové ihlice, pletacie stroje..)
- Aplikácia – tvorí súčasť výrobku, všetko čo podľa povahy výrobku k nemu patrí a nemôže byť oddelené bez toho, že by sa tým znehodnotil (gombíky, nažehlovačky, spony, brmbolce)
- Objednávka – žiadosť o vyhotovenie a dodanie tovaru s dopredu vymienenými vlastnosťami
- Zákazník – objednávateľ, osoba, ktorá objednáva tovar na osobné účely (pre ktorú je zhotovená objednávka)

Kód

Dedenie

V projekte sa nachádzajú tieto hierarchie dedenia:

- Prvá hierarchia dedenia je v balíku **employees**, kde je nasledovné **trojvrstvové dedenie**:

  - **Zamestnanci**
    - **Zamestnanec** (implements Zamestnanci)
      - **Manazer** (extends Zamestnanec)
      - **ZamestnanecVyroby** (extends Zamestnanec)

- Ďalšia hierarchia dedenia je v balíku **products**, a to týmto spôsobom:

  - **Odev**
    - **Sveter** (implements Odev)
      - **Vesta** (extends Sveter)

- Tretia hierarchia dedenia sa nachádza v balíku **machines**:

  - **Stroj**
    - **StrojNaSkladanie** (implements Stroj)
    - **StrojNaZehlenie** (implements Stroj)
      - **NaparovaciStrojNaZehlenie** (extends StrojNaZehlenie)

Rozhrania

V projekte používam rozhranie **Zamestnanci**, ako predpis pre všetky osoby kategórie zamestnanec (class Zamestnanec).

Rovnako používam rozhranie **Stroj** pre všetky triedy strojov (Class StrojNaSkladanie, Class StrojNaZehlenie), a rozhranie **Odev** pre všetky výrobky (class Sveter, class Vesta).

Polymorfizmus – prekonávanie

Prekonávanie metód nastáva v:

- class **Manazer**, metóda **odchodZPrace( )** okrem zmeny parametra vPraci, obsahuje metódu hromadnyOdchod( ), ktorá nastaví dochádzku všetkých podriadených zamestnancov manažéra na false, pretože ak nie je v práci vedúci manažér, jeho podriadení nemôžu pracovať.

```
public void odchodZPrace () { // override Zamestnanec
    this.hromadnyOdchod();
    this.vPraci = false;
}
```

- class **Zamestnanec**, metódy **prichodDoPrace( )** a **odchodZPrace( )** – tieto metódy dedí z interface Zamestnanci, a sú doplnené o nastavenie parametra boolean vPraci na true alebo false.

```
public void prichodDoPrace() { // override Zamestnanci
    this.vPraci = true;
}

public void odchodZPrace() { // override Zamestnanci
    this.vPraci = false;
}
```

- balík **machines**: všetky stroje obsahujú metódu **uprav( )**, a každá produkt upraví iným spôsobom (vyžehlenie, poskladanie, naparenie..)

```
public Sveter uprav (Sveter sveter) {
    sveter = poskladaj (sveter);
    return sveter;
}

public Vesta uprav (Vesta vesta) {
    vesta = poskladaj (vesta);
    return vesta;
}
```

```
public Sveter uprav (Sveter sveter) {
    sveter = vyzehli(sveter);
    return sveter;
}

public Vesta uprav (Vesta vesta) {
    vesta = vyzehli(vesta);
    return vesta;
}
```

- metóda **spracujObjednavky( )** v triede **Manazer** (rozdelenie úloh medzi zamestnancov):

```java
prebrané objednávky dalej rozdelí medzi svojich podriadených zamestnancov
public void spracujObjednavky () {
    int index;
    int j = 0;
    for (int i = (ulohy.size() - 1); i >= 0; i--) {

        if (i == 0) { index = 0; }
        else {
            index = i % this.pocetZamestnancov;
        }
        ObjednavkaZakaznika objednavka = this.ulohy.get(j);
        // predanie objednavky manazerovi
        this.getZamestnanci().get(index).preberObjednavku(objednavka);
        j++;
    }

    // odstranenie z aktivnych
    for (int i = (ulohy.size() - 1); i >= 0; i--) {...}

}
```

- metóda **spracujObjednavky( )** v triede **ZamestnanecVyroby** (vytvorenie produktov):

```java
vytvorenie finálnych produktov podľa zoznamu objednávok, ktoré má daný zamestnanec za úlohu vytvoriť
public void spracujObjednavky () {
    String velkost;
    String polozka;
    boolean damske;
    String material;
    String farba;

    for (int i = 0; i < this.ulohy.size(); i++) {

        for (int j = 0; j < this.ulohy.get(i).getPolozky().size(); j++) {

            // ziskanie parametrov
            velkost = this.ulohy.get(i).getPolozky().get(j).getVelkost();
            material = this.ulohy.get(i).getPolozky().get(j).getMaterial();
            farba = this.ulohy.get(i).getPolozky().get(j).getFarba();
            if (this.ulohy.get(i).getPolozky().get(j).getDamske() == 1) {...} else {...}

            //factory
            switch (polozka = this.ulohy.get(i).getPolozky().get(j).getNazovPolozky()) {
                case "vesta": // ak vyraba vestu
                    Vesta vesta = (Vesta)sijaciStroj.vyrobOdev(velkost, damske, material, polozka);
                    vesta.getMaterialVesty().setFarba(farba);
                    this.vesty.add(vesta);
                    vyrobeneVesty++;
                    break;
                case "sveter": // ak vyraba sveter
                    Sveter sveter = (Sveter)sijaciStroj.vyrobOdev(velkost, damske, material, polozka);
                    sveter.getMaterialSvetra().setFarba(farba);
                    this.svetre.add(sveter);
                    vyrobeneSvetre++;
                    break;
                default:
                    break;
            }
        }
    }

    // odstranenie hotových uloh
    for (int i = (ulohy.size() - 1); i >= 0; i--) {...}
}
```

Preťažovanie metód

Preťažovanie metód nastáva v:

- class **Manazer**, metódy **vykonajDochadzku( )** pre zadanie dochádzky všetkých zamestnancov, a **vykonajDochadzku(String meno)** pre vykonanie dochádzky konkrétneho zamestnanca, pre uľahčenie práce a šetrenie času
- v balíku **machines**, class StrojNaZehlenie, NaparovaciStrojNaZehlenie aj StrojNaSkladanie: **uprav(Sveter sveter)** a **uprav(Vesta vesta)**; a tiež jednotlivé metódy **vyzehli(Vesta vesta)**, **vyzehli(Sveter sveter)**...

Zapuzdrenie

Účelom zapuzdrenia je oddeliť rozhranie s kontraktom abstrakcie od jej implementácie. Trieda je „black box“, všetko je schované za public metódami. Použité sú modifikátory viditeľnosti (inštančné premenné sú private), prístup je spravovaný cez „gettre“ a „settre“.

Príklad:

Manazer

- parametre
  - → **private** ArrayList<ZamestnanecVyroby> zamestnanci
  - → **private** ArrayList<ObjednavkaZakaznika> ulohy
  - → **private** int pocetZamestnancov
- get / set
  - → **public** float getMzda( )
  - → **public** String getMeno( )
  - → **public** ArrayList<ZamestnanecVyroby> getZamestnanci( )
  - → **public** int getPocetZamestnancov( )
  - → **public** ArrayList<ObjednavkaZakaznika> getUlohy( )
  - → **public** void setZamestnanci(..)
  - → **public** void setMeno(..)
  - → **public** void setMzda(..)
  - → **public** void setVPraci(..)
  - → **public** void setPocetZamestnancov(..)
  - → **publiv** void setUlohy(..)

Zamestnanec

- parametre
  - → **protected** String meno
  - → **protected** boolean vPraci
  - → **protected** float mzda
- get / set
  - → **public** String getMeno( )
  - → **public** float getMzda( )
  - → **public** void setMeno(..)
  - → **public** void setMzda(..)
  - → **public** void setVPraci(..)

Agregácia

Agregácia nastáva v týchto prípadoch:

- **Sveter**
  - → private **Material** materialSvetra
- **Vesta**
  - → private **Material** materialVesty
- **Prevadzka**
  - → private **StrojNaSkladanie** strojNaSkladanie
  - → private **StrojNaZehlenie** strojNaZehlenie
  - → private **NaparovaciStrojNaZehlenie** naparovaciStrojNaZehlenie
  - → private **Sklad** sklad
  - → private ArrayList<Manazer> manazeri
  - → private ArrayList<ZamestnanecVyroby> zamestnanci
  - → private ArrayList<ObjednavkaZakaznika> aktivneObjednavky
- **Manazer**
  - → private ArrayList<ZamestnanecVyroby> zamestnanci
- **ObjednavkaZakaznika**
  - → private **InformacieZakaznika** informacie
  - → private ArrayList<PolozkaObjednavky> polozky

Organizácia kódu, balíky (Model, View, Controller)

Projekt sa skladá z týchto balíkov:

- classes
  - employees
    - Manazer
    - Zamestnanci
    - Zamestnanec
    - ZamestnanecVyroby
  - machines
    - NaparovaciStrojNaZehlenie
    - Stroj
    - StrojNaSkladanie
    - StrojNaZehlenie
  - orders
    - InformacieZakaznika
    - ObjednavkaZakaznika
    - PolozkaObjednavky
  - products
    - Material
    - Odev
    - Sveter
    - Vesta
  - warehouse
    - Prevadzka
    - Sklad
- core
  - DataHandler
  - Funkcionalita
  - Main
- messages
  - ProgramoveVypisy

Balík **classes** obsahuje všetky triedy, predstavuje Model. Balík **core** obsahuje hlavnú funkcionalitu programu, predstavuje Controller. Balík **messages** spravuje výstupy pre užívateľa, predstavuje View.

Návrhový vzor Singelton

Tento vzor je tvorený triedou **Prevadzka**, ktorá sa stará o to, aby jej inštancia existovala iba jedenkrát.

class **Prevadzka** implements **Serializable** {

```
        private static Prevadzka prevadzka = null;
        private Prevadzka (  ) { ......}
        public static Prevadzka getInstance(  ) {
                if (prevadzka == null) { prevadzka = new Prevadzka(  ); }
                return prevadzka;
        }
}
```

Návrhový vzor Factory

V našom programe sme potrebovali inštanciu (Vesta, Sveter), ktorú sme ale nedokázali spraviť samy. Zavolali sme teda Factory(SijaciStroj) – továreň – ktorá nám túto inštanciu vyrobila a dodala.

class **SijaciStroj**:

SijaciStroj je factory, ktorý vráti objekt na ktorý odkazujeme pomocou rozhrania Odev. Je využívaný v triede ZamestnanecVyroby.

```
public class SijaciStroj {

    public Odev vyrobOdev(String velkost, boolean damske, String material, String typ){
        if(typ == null){
            return null;
        }
        if(typ.toLowerCase().equals("sveter")){
            Thread t = new Thread (ZamestnanecVyroby::ThreadStatus);
            t.start();
            Sveter sveter = new Sveter(velkost, damske, material);
            sveter.setVyslednaCena(2*(sveter.getCenaMaterialu()));
            return sveter;
        }
        if (typ.toLowerCase().equals("vesta")){
            Thread t = new Thread (ZamestnanecVyroby::ThreadStatus);
            t.start();
            Vesta vesta = new Vesta(velkost, damske, material);
            vesta.setVyslednaCena(2*(vesta.getCenaMaterialu()));
            return vesta;
        }
        return null;
    }
```

Generičnosť

Pomocou Generics je možné vytvoriť triedy, ktoré pracujú s rôznymi dátovými typmi. V tomto projekte sa využíva generická trieda **InformacieZakaznika <M, A, E, P>**, kde meno, adresa a email sú predstavované dátovým typom String, a PSČ typom integer.

```
public class InformacieZakaznika <M, A, E, P> {

    M meno;
    A adresa;
    E email;
    P psc;
```

```
public InformacieZakaznika vypln () {
    Scanner keyboard = new Scanner(System.in);

    System.out.println(ProgramoveVypisy.infoMeno);
    String meno = keyboard.nextLine();
    System.out.println(ProgramoveVypisy.infoAdresa);
    String adresa = keyboard.nextLine();
    System.out.println(ProgramoveVypisy.infoPSC);
    int psc = keyboard.nextInt();

    System.out.println(ProgramoveVypisy.infoEmail);
    String email = "";

    int kontrola = 0;

    // vlastna vynimka
    do {...} while (kontrola == 1);

    this.vyplnene = 1;

    InformacieZakaznika <String, String, String, Integer> informacie = new InformacieZakaznika <String, String, String, Integer>

    return informacie;
}
```

Vlastná výnimka

V programe mám vytvorenú vlastnú výnimku (class **nespravnyEmailException**) na kontrolu správnosti formátu emailu zadaného zákazníkom pri objednávaní tovaru (použitá v triede **InformacieZakaznika**).

```
// vlastna vynimka
do {
    kontrola = 0;
    try {
        email = keyboard.nextLine();
        this.skontrolujEmail(email);
    } catch (NespravnyEmailException e) {
        kontrola = 1;

        if (email.length() > 0) {
            System.out.println(ProgramoveVypisy.errorColor + ProgramoveVypisy.errorEmail + ProgramoveVypisy.white);
        }
    }

    if (email.length() > 0 && kontrola == 1) {
        System.out.println(ProgramoveVypisy.infoEmail);
    }

} while (kontrola == 1);
```

```
public void skontrolujEmail (String email) throws NespravnyEmailException {
    int pocetZavinacov = 0;
    String nedovoleneZnaky = "#|€}{&#>đĐ[]!§()?";

    for (int i = 0; i < email.length(); i++) {
        if (email.charAt(i) == '@') {
            pocetZavinacov++;
        }

        for (int j = 0; j < nedovoleneZnaky.length(); j++) {
            if (nedovoleneZnaky.charAt(j) == email.charAt(i)) {
                throw new NespravnyEmailException("Email obsahuje nedovolené znaky");
            }
        }
    }

    if (pocetZavinacov > 1) {
        throw new NespravnyEmailException("Zadaný email nie je v správnom tvare.");
    }

    if (email.contains(".com") | email.contains(".sk") | email.contains(".cz")) { } else {
        throw new NespravnyEmailException("Zadaný email neobsahuje správny indikátor domény.");
    }
}
```

Kontrola prebieha pri každej novej objednávke.

Vhniezdená trieda

Vhniezdené triedy umožňujú logicky zoskupovať triedy, ktoré sa používajú iba na jednom mieste, čo zvyšuje použitie zapuzdrenia a vytvára čitateľnejší kód. V tomto projekte je vhniezdená trieda využitá v triede **NaparovaciStrojNaZehlenie**, a obsahuje statickú triedu **Naparovac**, ktorá predstavuje súčasť naparovacieho stroju, ktorá umožňuje naparovanie. Tento komponent sám o sebe nemôže existovať.

```
private static class Naparovac {
    private static final double indexSpokojnosti  = 2.5;

    private static Sveter napar(Sveter sveter) {
        sveter.setBonusKSpokojnosti(sveter.getBonusKSpokojnosti() + indexSpokojnosti);
        return sveter;
    }

    private static Vesta napar(Vesta vesta) {
        vesta.setBonusKSpokojnosti(vesta.getBonusKSpokojnosti() + indexSpokojnosti);
        return vesta;
    }
}

public Sveter uprav (Sveter sveter) {

    sveter = Naparovac.napar(sveter);
    sveter = vyzehli(sveter);

    return sveter;
}
```

Method References & Multithreading

V triede **SijaciStroj** používame preddefinované funkčné rozhranie Runnable na odkazovanie na statickú metódu. Rovnako sa pri tom využívajú vlákna (threads) pri vytváraní produktov. Zamestnanec takto dáva vedieť stav výroby objednávky.

```
public Odev vyrobOdev(String velkost, boolean damske, String material, String typ){
    if(typ == null){
        return null;
    }
    if(typ.toLowerCase().equals("sveter")){
        Thread t = new Thread (ZamestnanecVyroby::ThreadStatus);
        t.start();
        Sveter sveter = new Sveter(velkost, damske, material);
        sveter.setVyslednaCena(2*(sveter.getCenaMaterialu()));
        return sveter;
    }
    if (typ.toLowerCase().equals("vesta")){
        Thread t = new Thread (ZamestnanecVyroby::ThreadStatus);
        t.start();
        Vesta vesta = new Vesta(velkost, damske, material);
        vesta.setVyslednaCena(2*(vesta.getCenaMaterialu()));
        return vesta;
    }
    return null;
```

Default Method Implementation

V rozhraní Zamestnanec je použitá implicitná implementácia metódy **pracovnaPozicia( )**, ktorá pomocou RTTI zistí, o aký objekt sa jedná.

```
default void pracovnaPozicia () {

    if (this instanceof ZamestnanecVyroby) {
        System.out.print("zamestnanec výroby ");
    }

    if (this instanceof Manazer) {
        System.out.print("manažér\n");
    }

}
```

RTTI (instanceof)

Java instanceof operátor sa používa na testovanie, či je objekt inštancií špecifikovaného typu (triedy alebo podtriedy alebo rozhrania).

Instanceof v jave je tiež známy ako *operátor porovnania* typov, pretože porovnáva inštanciu s typom. Vráti hodnotu true alebo false. Ak použijeme operátor instanceof s ľubovoľnou premennou, ktorá má nulovú hodnotu, vráti hodnotu false.

Okrem vyššie spomenutého prípadu, v tomto projekte sa používa aj na kontrolu existencie objednávky (class Funkcionalita):

```
case 4: // zobrazenie objednavky
    if (!(objednavka instanceof ObjednavkaZakaznika)) { // ak neeistuje
        System.out.println(ProgramoveVypisy.errorColor + ProgramoveVypisy.caseZakaznikNeexistuje
        break;
    }
    System.out.println(ProgramoveVypisy.caseZakaznikZobrazenie);
    objednavka.vypisObjednavku();
    break;
```

Funkcionalita

Funkcionalita programu sa delí podľa typu osoby, ktorá program spúšťa – každá má vlastné oprávnenia.

Switch: caseOperator

„Operátor“ je osoba zodpovedná za existenciu prevádzky, je akoby „vyššou mocou“, ktorá má najväčšie oprávnenia.

1. Vytvorenie prevádzky

Ako prvé pred prácou s programom je potrebné vytvoriť inštanciu prevádzky, preto že bez nej nemôžu byť spustiteľné ďalšie prvky funkcionality. Za celý beh programu existuje iba jediná inštancia prevádzky.

2. Zamestnanie manažérov

Je to ďalší dôležitý krok – ak prevádzka nemá manažérov, nemôže mať ani zamestnancov výroby, a teda nie je schopná vykonávať výrobný proces. Je možné zamestnať ľubovoľný počet manažérov.

Ak prevádzka neexistuje, nie je možné zamestnať manažérov.

3. Zamestnanie zamestnancov výroby

Pre každého z manažérov je možné zamestnať ľubovoľný počet jemu podriadených zamestnancov výroby. Toto oprávnenie má aj sám manažér.

Ak nie sú zamestnaní žiadni manažéri, nie je možné zamestnať ani zamestnancov výroby.

4. Kontrola pracoviska

Ide o hlásenie dochádzky celého personálu prevádzky – manažérov a im podriadených zamestnancov.

Ak nie sú zamestnaní žiadni manažéri, nie je možné vykonať kontrolu prevádzky.

5. Vypísanie obsahu skladu

Pokiaľ sa v sklade nachádzajú už vyrobené produkty čakajúce na odoslanie, vypíšu sa.

6. Vypísanie zisku prevádzky

Ide o zisk z predaja produktov, s už odpočítanými nákladmi na materiál.

7. Odoslanie produktov zákazníkom

Funkcia odošle zákazníkom produkty, ktoré si objednali, ktoré sú už vytvorené a čakajú v sklade na odoslanie.

Switch: caseManazer

Pre prístup k oprávneniam manažéra je potrebné identifikovať sa ako jeden z existujúcich manažérov. Musia teda už byť vytvorení manažéri v caseOperator. Tým, že sa manažér prihlási, sa mu automaticky vykoná dochádzka do práce.

1. Vykonanie dochádzky podriadených zamestnancov

Táto funkcia umožní manažérovi zadať dochádzku všetkým jeho podriadeným zamestnancom.

Pre vykonanie dochádzky zamestnancov musia existovať zamestnanci.

2. Vykonanie dochádzky konkrétneho zamestnanca

Táto funkcia tiež slúži na zadanie dochádzky, ale v prípade ak ide iba o konkrétneho zamestnanca, aby sa šetril manažérovi čas tým, že nemusí robiť celú dochádzku odznova.

3. Zamestnanie nových zamestnancov výroby

Manažér má oprávnenie zamestnať pod seba (ďalších) sebe podriadených zamestnancov výroby.

4. Vykonanie hlásenia o zamestnancoch a dochádzke

Funkcia vypíše meno manažéra a všetkých jemu podriadených zamestnancov, spolu s tým, či sa nachádzajú v práci, alebo nie.

5. Vykonanie hlásenia o objednávkach

Pokiaľ má manažér pridelené úlohy, ktoré spravuje, vypíšu sa.

6. Rozdelenie úloh zamestnancom

Pokiaľ má manažér pridelené úlohy, ktoré spravuje, rozdelí ich medzi svojich podriadených zamestnancov, aby ich mohli splniť.

Switch: caseZamestnanec

Pre využitie oprávnení zamestnanca je potrebné, aby už bola vytvorená prevádzka a zamestnaní manažéri. Pre prístup k oprávneniam je potrebné identifikovať sa ako jeden z existujúcich zamestnancov. Tým, že sa zamestnanec prihlási, sa mu automaticky vykoná dochádzka do práce.

1. Vykonanie hlásenia o úlohách

Pokiaľ má zamestnanec pridelené úlohy, ktoré má vykonať, vypíšu sa.

2. Vytvorenie produktov

Pokiaľ má zamestnanec pridelené úlohy, ktoré má vykonať, vyrobí podľa nich produkty.

Switch: caseZakaznik

Pre využitie oprávnení zákazníka je potrebné, aby už bola vytvorená prevádzka a zamestnaní manažéri. Po odhlásení sa zákazníka sa automaticky nové objednávky (aktívne objednávky v prevádzke) prerozdelia medzi manažérov.

1. Pridanie výrobku do objednávky

Pri vytváraní / pridávaní položiek do objednávky sa postupne program spýta na druh a parametre želaného produktu. Pokiaľ program zadaný vstup nerozpozná ako možný druh produktu, do objednávky nebude pridaný a zákazníka na to upozorní.

2. Odstránenie produktu z objednávky

Táto funkcia ešte nie je implementovaná.

3. Potvrdenie objednávky

Pri potvrdzovaní objednávky program zobrazí výslednú čiastku, a spýta sa, či si zákazník vážne praje objednávku odoslať. Po potvrdení (a vyplnení informácií o zákazníkovi) sa objednávka automaticky presunie medzi aktívne objednávky do prevádzky.

Pre potvrdenie objednávky musí objednávka existovať, teda musí byť pridaná aspoň jedna položka objednávky.

4. Zobrazenie aktuálnej objednávky

Funkcia zobrazí všetky položky objednávky, spolu s ich cenou a s výslednou cenou celej objednávky.

Pre zobrazenie objednávky musí objednávka existovať, teda musí byť pridaná aspoň jedna položka objednávky.

Ukážka chodu programu

```
Spustit program ako:
-> operator (1)
-> manažér (2)
-> zamestnanec (3)
-> zákazník (4)
-> pre ukončenie programu stlačte 0
1
Spustený mód operátor
-> vytvorenie prevádzky (1)
-> zamestnanie manažérov (2)
-> zamestnanie zamestnancov výroby (3)
-> kontrola pracovníkov (4)
-> vypíš obsah skladu (5)
-> vypíš zisk prevádzky (6)
-> odošli produkty zákazníkom (7)
-> pre ukončenie módu operátor stlačte 0
1
Operácia prebehla úspešne.

-> vytvorenie prevádzky (1)
-> zamestnanie manažérov (2)
-> zamestnanie zamestnancov výroby (3)
-> kontrola pracovníkov (4)
-> vypíš obsah skladu (5)
-> vypíš zisk prevádzky (6)
-> odošli produkty zákazníkom (7)
-> pre ukončenie módu operátor stlačte 0
2
Koľko manažérov chcete zamestnať?
1
Zamestnávanie manažérov:
Zadaj meno manažéra č.1 :
Richard
Operácia prebehla úspešne.
```

```
-> vytvorenie prevádzky (1)
-> zamestnanie manažérov (2)
-> zamestnanie zamestnancov výroby (3)
-> kontrola pracovníkov (4)
-> vypíš obsah skladu (5)
-> vypíš zisk prevádzky (6)
-> odošli produkty zákazníkom (7)
-> pre ukončenie módu operátor stlačte 0
3
Zamestnávanie pracovníkov výroby:
Koľko (ďalších) podriadených má mať manažér Richard ?
2
Zadaj meno zamestnanca č.1 :
Andrej
Zadaj meno zamestnanca č.2 :
Ondrej
Operácia prebehla úspešne.
```

```
-> vytvorenie prevádzky (1)
-> zamestnanie manažérov (2)
-> zamestnanie zamestnancov výroby (3)
-> kontrola pracovníkov (4)
-> vypíš obsah skladu (5)
-> vypíš zisk prevádzky (6)
-> odošli produkty zákazníkom (7)
-> pre ukončenie módu operátor stlačte 0
4

Hlásenie personálu prevádzky:
Manažér Richard nie je v práci
    Hlásenie: volám sa Richard, som manažér, moji podriadení zamestnanci sú:
    zamestnanec výroby Andrej, nie je prítomný
    zamestnanec výroby Ondrej, nie je prítomný

-> vytvorenie prevádzky (1)
-> zamestnanie manažérov (2)
-> zamestnanie zamestnancov výroby (3)
-> kontrola pracovníkov (4)
-> vypíš obsah skladu (5)
-> vypíš zisk prevádzky (6)
-> odošli produkty zákazníkom (7)
-> pre ukončenie módu operátor stlačte 0
5
Sklad je momentálne prázdny

-> vytvorenie prevádzky (1)
-> zamestnanie manažérov (2)
-> zamestnanie zamestnancov výroby (3)
-> kontrola pracovníkov (4)
-> vypíš obsah skladu (5)
-> vypíš zisk prevádzky (6)
-> odošli produkty zákazníkom (7)
-> pre ukončenie módu operátor stlačte 0
6
Zisk prevádzky je 0.0 €
```

```
-> vytvorenie prevádzky (1)
-> zamestnanie manažérov (2)
-> zamestnanie zamestnancov výroby (3)
-> kontrola pracovníkov (4)
-> vypíš obsah skladu (5)
-> vypíš zisk prevádzky (6)
-> odošli produkty zákazníkom (7)
-> pre ukončenie módu operátor stlačte 0
0
```

```
Spustit program ako:
-> operator (1)
-> manažér (2)
-> zamestnanec (3)
-> zákazník (4)
-> pre ukončenie programu stlačte 0
2

Spustený mód manažér
Identifikujte sa prosím
-> Richard (0)
0
Vitajte naspäť Richard

-> vykonanie dochádzky všetkých zamestnancov (1)
-> doplnenie dochádzky (2)
-> zamestnanie zamestnancov (3)
-> vykonanie hlásenia o dochádzke (4)
-> vykonanie hlásenia o objednávkach (5)
-> rozdel úlohy zamestnancom (6)
-> pre ukončenie módu manažér stlačte 0
4

Vaše hlásenie znie:
Hlásenie: volám sa Richard, som manažér, moji podriadení zamestnanci sú:
    zamestnanec výroby Andrej, nie je prítomný
    zamestnanec výroby Ondrej, nie je prítomný

-> vykonanie dochádzky všetkých zamestnancov (1)
-> doplnenie dochádzky (2)
-> zamestnanie zamestnancov (3)
-> vykonanie hlásenia o dochádzke (4)
-> vykonanie hlásenia o objednávkach (5)
-> rozdel úlohy zamestnancom (6)
-> pre ukončenie módu manažér stlačte 0
5
Momentálne nemáte pridelené žiadne objednávky
```

```
-> vykonanie dochádzky všetkých zamestnancov (1)
-> doplnenie dochádzky (2)
-> zamestnanie zamestnancov (3)
-> vykonanie hlásenia o dochádzke (4)
-> vykonanie hlásenia o objednávkach (5)
-> rozdel úlohy zamestnancom (6)
-> pre ukončenie módu manažér stlačte 0
1
```

```
Zadajte dochádzku
Je prítomný zamestnanec Andrej ? (a/n)
a
Zamestnanec sa dostavil do práce
Je prítomný zamestnanec Ondrej ? (a/n)
a
Zamestnanec sa dostavil do práce

-> vykonanie dochádzky všetkých zamestnancov (1)
-> doplnenie dochádzky (2)
-> zamestnanie zamestnancov (3)
-> vykonanie hlásenia o dochádzke (4)
-> vykonanie hlásenia o objednávkach (5)
-> rozdel úlohy zamestnancom (6)
-> pre ukončenie módu manažér stlačte 0
4

Vaše hlásenie znie:
Hlásenie: volám sa Richard, som manažér, moji podriadení zamestnanci sú:
    zamestnanec výroby Andrej, je prítomný
    zamestnanec výroby Ondrej, je prítomný

-> vykonanie dochádzky všetkých zamestnancov (1)
-> doplnenie dochádzky (2)
-> zamestnanie zamestnancov (3)
-> vykonanie hlásenia o dochádzke (4)
-> vykonanie hlásenia o objednávkach (5)
-> rozdel úlohy zamestnancom (6)
-> pre ukončenie módu manažér stlačte 0
0

Spustit program ako:
-> operator (1)
-> manažér (2)
-> zamestnanec (3)
-> zákazník (4)
-> pre ukončenie programu stlačte 0
4
```

```
Spustený mód zákazník
-> pridanie do objednávky (1)
-> odstránenie z objednávky (2)
-> potvrdenie objednávky (3)
-> zobrazenie objednávky (4)
-> pre ukončenie módu zákazník stlačte 0
1
```

```
Zadajte typ tovaru ktorý si želáte objednať (vesta, sveter):
sveter
Zadajte želanú farbu tovaru:

Zadajte želaný materiál (bavlna, hodvab, moher, merino):
zelezo
Zadajte veľkosť tovaru (XS, S, M, L, XL):
S
Má byť želaný produkt dámsky (1) alebo pánsky (0) ?
0
CHYBA: Nebol zadaný správny vstup.

-> pridanie do objednávky (1)
-> odstránenie z objednávky (2)
-> potvrdenie objednávky (3)
-> zobrazenie objednávky (4)
-> pre ukončenie módu zákazník stlačte 0
1

Zadajte typ tovaru ktorý si želáte objednať (vesta, sveter):
sveter
Zadajte želanú farbu tovaru:
cierny
Zadajte želaný materiál (bavlna, hodvab, moher, merino):
bavlna
Zadajte veľkosť tovaru (XS, S, M, L, XL):
S
Má byť želaný produkt dámsky (1) alebo pánsky (0) ?
1
Operácia prebehla úspešne.

-> pridanie do objednávky (1)
-> odstránenie z objednávky (2)
-> potvrdenie objednávky (3)
-> zobrazenie objednávky (4)
-> pre ukončenie módu zákazník stlačte 0
4
```

```
Vaša objednávka je nasledujúca:
Typ: sveter, materiál: bavlna, farba: cierny, veľkosť: S, strih: dámsky, cena: 70.0 EUR
SPOLU: 70.0 EUR

-> pridanie do objednávky (1)
-> odstránenie z objednávky (2)
-> potvrdenie objednávky (3)
-> zobrazenie objednávky (4)
-> pre ukončenie módu zákazník stlačte 0
1
```

```
Zadajte typ tovaru ktorý si želáte objednať (vesta, sveter):
vesta
Zadajte želanú farbu tovaru:
hneda
Zadajte želaný materiál (bavlna, hodvab, moher, merino):
hodvab
Zadajte veľkosť tovaru (XS, S, M, L, XL):
M
Má byť želaný produkt dámsky (1) alebo pánsky (0) ?
0
Operácia prebehla úspešne.

-> pridanie do objednávky (1)
-> odstránenie z objednávky (2)
-> potvrdenie objednávky (3)
-> zobrazenie objednávky (4)
-> pre ukončenie módu zákazník stlačte 0
4

Vaša objednávka je nasledujúca:
Typ: sveter, materiál: bavlna, farba: cierny, veľkosť: S, strih: dámsky, cena: 70.0 EUR
Typ: vesta, materiál: hodvab, farba: hneda, veľkosť: M, strih: pánsky, cena: 360.0 EUR
SPOLU: 430.0 EUR

-> pridanie do objednávky (1)
-> odstránenie z objednávky (2)
-> potvrdenie objednávky (3)
-> zobrazenie objednávky (4)
-> pre ukončenie módu zákazník stlačte 0
3
Výsledná cena: 430.0 EUR

Želáte si Vašu objednávku potvrdiť a odoslať? (a/n)
a
Zadajte vaše meno:
Emma
Zadajte vašu adresu:
Adresa v Bratislave 13
Zadajte vaše poštové smerovacie číslo:
83102
Zadajte vašu e-mailovú adresu:
nie
CHYBA: Email nebol zadaný v správnom tvare.
Zadajte vašu e-mailovú adresu:
email@email
CHYBA: Email nebol zadaný v správnom tvare.
Zadajte vašu e-mailovú adresu:
email@email.sk
```

```
Informácie pre doručenie:
Emma, email@email.sk, Adresa v Bratislave 13, 83102

Želáte si Vašu objednávku potvrdiť a odoslať? (a/n)
a
Operácia prebehla úspešne.

-> pridanie do objednávky (1)
-> odstránenie z objednávky (2)
-> potvrdenie objednávky (3)
-> zobrazenie objednávky (4)
-> pre ukončenie módu zákazník stlačte 0
0

Spustit program ako:
-> operator (1)
-> manažér (2)
-> zamestnanec (3)
-> zákazník (4)
-> pre ukončenie programu stlačte 0
2

Spustený mód manažér
Identifikujte sa prosím
-> Richard (0)
0
Vitajte naspäť Richard

-> vykonanie dochádzky všetkých zamestnancov (1)
-> doplnenie dochádzky (2)
-> zamestnanie zamestnancov (3)
-> vykonanie hlásenia o dochádzke (4)
-> vykonanie hlásenia o objednávkach (5)
-> rozdel úlohy zamestnancom (6)
-> pre ukončenie módu manažér stlačte 0
5
```

```
Vaše hlásenie znie:
Objednávky ktoré spravujem sú:
Typ: sveter, materiál: bavlna, farba: cierny, veľkosť: S, strih: dámsky, cena: 70.0 EUR
Typ: vesta, materiál: hodvab, farba: hneda, veľkosť: M, strih: pánsky, cena: 360.0 EUR
SPOLU: 430.0 EUR

-> vykonanie dochádzky všetkých zamestnancov (1)
-> doplnenie dochádzky (2)
-> zamestnanie zamestnancov (3)
-> vykonanie hlásenia o dochádzke (4)
-> vykonanie hlásenia o objednávkach (5)
-> rozdel úlohy zamestnancom (6)
-> pre ukončenie módu manažér stlačte 0
0
```

```
Spustit program ako:
-> operator (1)
-> manažér (2)
-> zamestnanec (3)
-> zákazník (4)
-> pre ukončenie programu stlačte 0
3

Spustený mód zamestnanec
Identifikujte sa prosím
-> Andrej (0 0)
-> Ondrej (0 1)
0 0
Vitajte naspäť Andrej

-> vykonanie hlásenia o úlohách (1)
-> začni výrobu produktov (2)
-> pre ukončenie módu zamestnanec stlačte 0
1
Momentálne nemáte pridelené žiadne objednávky

Vaše hlásenie znie:
Objednávky ktoré spravujem sú:
Momentálne nemáte žiadne aktívne objednávky. Počet už vyrobených svetrov: 0, a počet vyrobených viest: 0

-> vykonanie hlásenia o úlohách (1)
-> začni výrobu produktov (2)
-> pre ukončenie módu zamestnanec stlačte 0
0

Spustit program ako:
-> operator (1)
-> manažér (2)
-> zamestnanec (3)
-> zákazník (4)
-> pre ukončenie programu stlačte 0
2
```

```
Spustený mód manažér
Identifikujte sa prosím
-> Richard (0)
0
Vitajte naspäť Richard

-> vykonanie dochádzky všetkých zamestnancov (1)
-> doplnenie dochádzky (2)
-> zamestnanie zamestnancov (3)
-> vykonanie hlásenia o dochádzke (4)
-> vykonanie hlásenia o objednávkach (5)
-> rozdel úlohy zamestnancom (6)
-> pre ukončenie módu manažér stlačte 0
6
Operácia prebehla úspešne.
```

```
-> vykonanie dochádzky všetkých zamestnancov (1)
-> doplnenie dochádzky (2)
-> zamestnanie zamestnancov (3)
-> vykonanie hlásenia o dochádzke (4)
-> vykonanie hlásenia o objednávkach (5)
-> rozdel úlohy zamestnancom (6)
-> pre ukončenie módu manažér stlačte 0
2
Čiu dochádzku si želáte upraviť?
Ondrej
Je prítomný zamestnanec Ondrej ? (a/n)
n
Zamestnanec absentuje

-> vykonanie dochádzky všetkých zamestnancov (1)
-> doplnenie dochádzky (2)
-> zamestnanie zamestnancov (3)
-> vykonanie hlásenia o dochádzke (4)
-> vykonanie hlásenia o objednávkach (5)
-> rozdel úlohy zamestnancom (6)
-> pre ukončenie módu manažér stlačte 0
0

Spustit program ako:
-> operator (1)
-> manažér (2)
-> zamestnanec (3)
-> zákazník (4)
-> pre ukončenie programu stlačte 0
3

Spustený mód zamestnanec
Identifikujte sa prosím
-> Andrej (0 0)
-> Ondrej (0 1)
0 0
Vitajte naspäť Andrej
```

```
-> vykonanie hlásenia o úlohách (1)
-> začni výrobu produktov (2)
-> pre ukončenie módu zamestnanec stlačte 0
1

Vaše hlásenie znie:
Objednávky ktoré spravujem sú:
Typ: sveter, materiál: bavlna, farba: cierny, veľkosť: S, strih: dámsky, cena: 70.0 EUR
Typ: vesta, materiál: hodvab, farba: hneda, veľkosť: M, strih: pánsky, cena: 360.0 EUR
SPOLU: 430.0 EUR

-> vykonanie hlásenia o úlohách (1)
-> začni výrobu produktov (2)
-> pre ukončenie módu zamestnanec stlačte 0
2
Produkt sa vytvára...
Produkt sa vytvára...
Operácia prebehla úspešne.
```

```
-> vykonanie hlásenia o úlohách (1)
-> začni výrobu produktov (2)
-> pre ukončenie módu zamestnanec stlačte 0
0

Spustit program ako:
-> operator (1)
-> manažér (2)
-> zamestnanec (3)
-> zákazník (4)
-> pre ukončenie programu stlačte 0
1
Spustený mód operátor
-> vytvorenie prevádzky (1)
-> zamestnanie manažérov (2)
-> zamestnanie zamestnancov výroby (3)
-> kontrola pracovníkov (4)
-> vypíš obsah skladu (5)
-> vypíš zisk prevádzky (6)
-> odošli produkty zákazníkom (7)
-> pre ukončenie módu operátor stlačte 0
5
Obsah skladu je nasledovný:
Sveter, materiál: bavlna, farba: cierny, veľkosť: S, strih: dámsky, bonus spokojnosti: 4.5, cena bez bonusu: 70.0, cena s bonusom: 315.0
Vesta, materiál: hodvab, farba: hneda, veľkosť: M, strih: pánsky, bonus spokojnosti: 4.5, cena bez bonusu: 360.0, cena s bonusom: 1620.0

-> vytvorenie prevádzky (1)
-> zamestnanie manažérov (2)
-> zamestnanie zamestnancov výroby (3)
-> kontrola pracovníkov (4)
-> vypíš obsah skladu (5)
-> vypíš zisk prevádzky (6)
-> odošli produkty zákazníkom (7)
-> pre ukončenie módu operátor stlačte 0
6
Zisk prevádzky je 0.0 €
```

```
-> vytvorenie prevádzky (1)
-> zamestnanie manažérov (2)
-> zamestnanie zamestnancov výroby (3)
-> kontrola pracovníkov (4)
-> vypíš obsah skladu (5)
-> vypíš zisk prevádzky (6)
-> odošli produkty zákazníkom (7)
-> pre ukončenie módu operátor stlačte 0
7
Operácia prebehla úspešne.

-> vytvorenie prevádzky (1)
-> zamestnanie manažérov (2)
-> zamestnanie zamestnancov výroby (3)
-> kontrola pracovníkov (4)
-> vypíš obsah skladu (5)
-> vypíš zisk prevádzky (6)
-> odošli produkty zákazníkom (7)
-> pre ukončenie módu operátor stlačte 0
5
Sklad je momentálne prázdny
```

```
-> vytvorenie prevádzky (1)
-> zamestnanie manažérov (2)
-> zamestnanie zamestnancov výroby (3)
-> kontrola pracovníkov (4)
-> vypíš obsah skladu (5)
-> vypíš zisk prevádzky (6)
-> odošli produkty zákazníkom (7)
-> pre ukončenie módu operátor stlačte 0
6
Zisk prevádzky je 967.5 €
```

```
-> vytvorenie prevádzky (1)
-> zamestnanie manažérov (2)
-> zamestnanie zamestnancov výroby (3)
-> kontrola pracovníkov (4)
-> vypíš obsah skladu (5)
-> vypíš zisk prevádzky (6)
-> odošli produkty zákazníkom (7)
-> pre ukončenie módu operátor stlačte 0
1
CHYBA: Prevádzka už bola vytvorená

-> vytvorenie prevádzky (1)
-> zamestnanie manažérov (2)
-> zamestnanie zamestnancov výroby (3)
-> kontrola pracovníkov (4)
-> vypíš obsah skladu (5)
-> vypíš zisk prevádzky (6)
-> odošli produkty zákazníkom (7)
-> pre ukončenie módu operátor stlačte 0
4

Hlásenie personálu prevádzky:
Manažér Richard je v práci
    Hlásenie: volám sa Richard, som manažér, moji podriadení zamestnanci sú:
    zamestnanec výroby Andrej, je prítomný
    zamestnanec výroby Ondrej, nie je prítomný
```

Hlavné verzie na GitHub - e

Commits on Mar 17, 2021

- vytvorenie základu projektu,
- prvé triedy

Commits on Mar 18, 2021

- jemné úpravy

Commits on Mar 31, 2021

- základ funkcionality,
- pridanie produktu vesta,
- vytvorenie objednávok,
- zmena prevádzky na singleton,
- pridanie metód zamestnancom..

Commits on Apr 04, 2021

- pridanie metód prevádzke, manažérom a objednávke,
- aktualizácia funkcionality

Commits on Apr 07, 2021

- úprava všetkých tried a funkcionality,
- pridanie dokumentácie

Commits on Apr 11, 2021

- verzia pred prvým odovzdaním

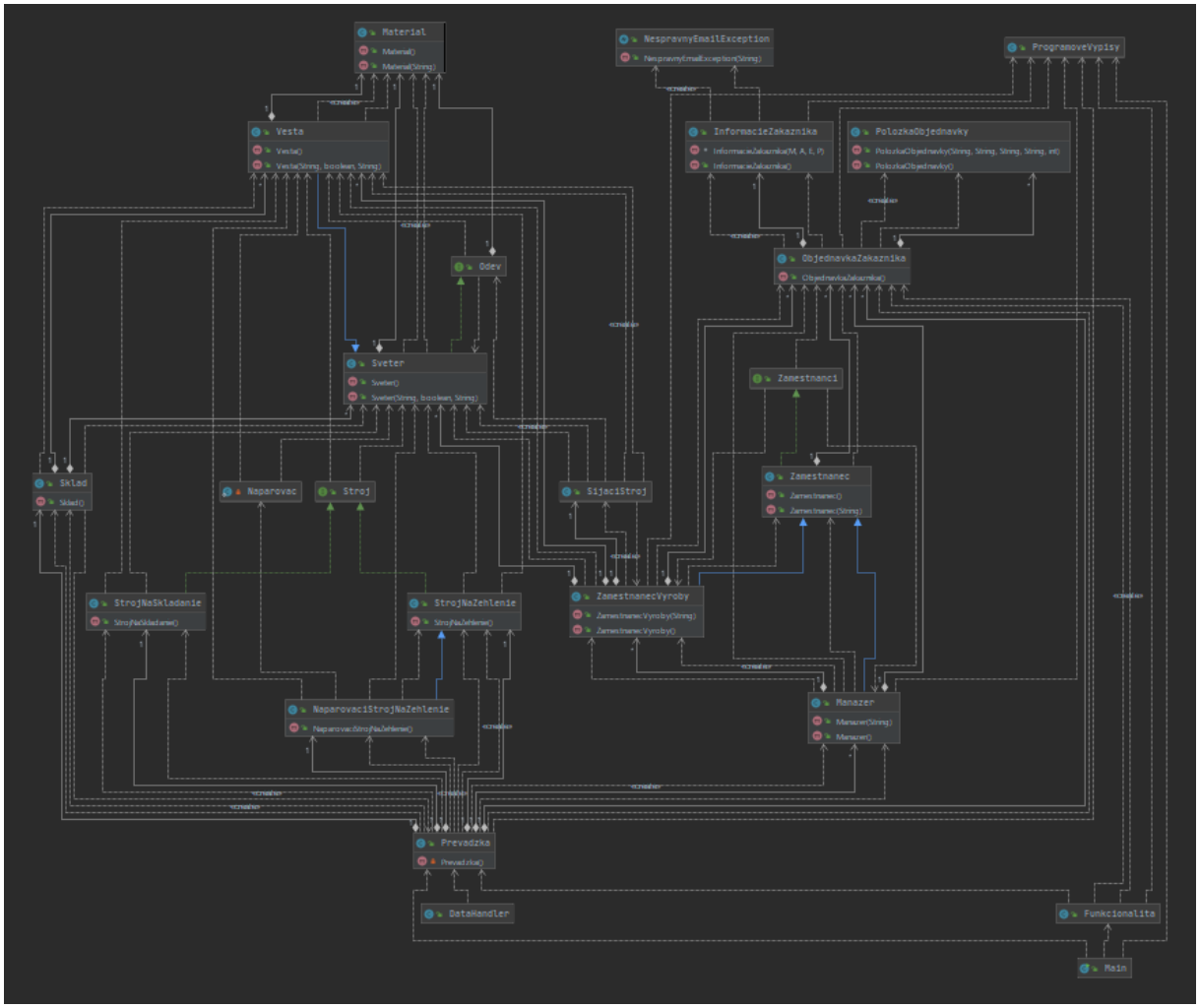
Commits on May 06, 2021

- finalizácia metód a funkcionality,
- pridanie kritérií

Commits on May 12, 2021

- doplnenie komentárov a súborov pre javadoc

UML diagram tried



Podrobnejší UML diagram je v priečinku na Github-e.

Záver

Softvér, tak ako bolo stanovené v ciele projektu, slúži na manažovanie procesov, ktoré sú spojené s výrobou a následným predajom handmade výrobkov lokálnou spoločnosťou.

Medzi tieto procesy patrí najmä správa inventáru spoločnosti, samotná výroba konečných produktov, rozdeľovanie úloh medzi zamestnancov a konečné odoslanie výrobku zákazníkovi. Tak isto bol dosiahnutý tok informácií medzi jednotlivými oddeleniami a kľúčovými zamestnancami. Zamestnanci pracujú na rôznych typoch pracovných pozícií, ich schopnosti a úlohy sa líšia. Výsledným produktom výrobného procesu je pletený výrobok (sveter alebo vesta), ktoré je možné vytvárať v rôznych variantoch (farba, materiál, strih..).

Funkcionalita je rozdelená pre jednotlivé druhy užívateľov pre optimalizáciu efektívnosti, prehľadnosti a napodobeniu reálneho procesu tvorenia objednávok a vytvárania produktov.

Systém neberie do úvahy čas potrebný na konečný export produktov zákazníkovi (sklad je uvoľnený momentom predaja). Pripodobňuje reálny proces výroby, výroba ani rozdeľovanie úloh neprebehne bez zapojenia sa zamestnanca. Softvér počíta náklady na materiál a zisk z predaja výrobkov.