Академија струковних студија Шумадија
Одсек Крагујевац

Студијски програм: Информатика

Предмет: Познавање пословних процеса

ПРЕДЛОГ РЕШЕЊА

-1.4-

Предметни наставник:

Саша Стаменовић

Студент:

Јована Томић 003/2020

Крагујевац 2022.

РАДНА ВЕРЗИЈА	ПРЕДАТО	ОПИС
1.0	30.3.2022.	Прва верзија предлога
1.1	12.5.2022.	Додата табела, прецизиран текст
1.2	6.6.2022.	Побољшан фонт
1.3	16.6.2022.	Исправљен садржај предлога решења
1.4	29.8.2022.	Детаљније објашњена очекивана сврха апликације

Contents

1. УВОД 4
2.0. ЗАХТЕВ НАБАВКЕ И ПОРУЏБЕНИЦЕ 6
3. ДОБАВЉАЧИ 7
4.ПРОИЗВОДИ 8
5.БАЗА ПОДАТАКА ЗАХТЕВА ЗА НАБАВКУ ПОРУЂБЕНИЦЕ 9
6.БАЗА ПОДАТАКА ДОБАВЉАЧА 10
7. БАЗА ПОДАТАКА ЗА ПРОИЗВОДЕ 11
4.4ПАКЕТ controller 12
4.5 ПАКЕТ aspect 13
5.0 ЗАКЉУЧАК 15

1. УВОД

Наша софтвер ће се базирати на олакшавању рада процеса набавке. Софтвер ће омогућити набавку одређених количина материјала по уговореној цени од одобрених добављача. Апликација ће обезбедити преглед захтева за набавку и распоређивање производа по добављачима за потреби набавке.

За израду ове апликације, користиће се интерна база података за складиштење података о добавњачима која ће бити у оквиру набавке. Такође, та база ће садржати информације о свим захтевима набавке и свим поруџбеницама. Постојаће екстерна база података за складиштење података о тренутном стању магацина из којег ће се извлачити информације о стању материјала.

1.1 ОПИС СОФТВЕРА

Софтвер ће функционисати тако што ће при отварању истог корисник моћи одмах да се пријави, уколико има налог.

Уколико корисник нема налог, а жели да се региструје мораће да креира нови налог.

Након креирања налога и пријављивања на сам софтвер корисник ће бити усмерен на део софтера за приступ прегледима којима располаже набавка, а то су:

- Преглед производа
- Преглед добављача
- Преглед поруџбеница
- Преглед захтева набавке

Преглед производа ће садржату преглед производа као и опцију за њихово брисање и уређивање.

Преглед добављача ће имати врло сличне функције, док ће се преглед поруџбеница и мало разликовати.

У прегледу поруџбеница ће се сем саме поруџбенице моћи видети и сви производи који је чине, такође овај преглед ће имати опцију за брисање и креирање нових.

Преглед захтева набавке ће бити јако сличан као и захтев поруџбеница, с тим што код креирања захтева набавке се може захтевати материјал од више добављача, док на поруџбеници имамо опцију за само једног добављача.

2.0. ЗАХТЕВ НАБАВКЕ И ПОРУЏБЕНИЦЕ

У нашој апликацији постоји доста опција за рад над захтевима за набавку и то:

- Преглед
- Додавање нових
- Ажурирање постојећих
- Брисање

Такође, у постојаће опција за преглед производа на основу изабраног захтева или саме поруџбенице.

3. ДОБАВЉАЧИ

Унутар апликације ће се наћи део интерфејс преко кога ће бити могуће видети све добављаче и њихове податке(Земља, Град, Улица...). Унутар тог интерфејса ће бити могуће додати, изменити или обрисати добављача у зависности од потребе предузећа.

4. ПРОИЗВОДИ

Унутар апликације постојаће посебан интерфејс за производе преко ког ће бити могуће видети све производе које предузеће користи уз њихове добављаче. Такође, постојаће опције за додавање нових производа, брисање производа или промену постојећих производа.

5. БАЗА ПОДАТАКА ЗАХТЕВА ЗА НАБАВКУ И ПОРУЏБЕНИЦЕ

Користићемо SQLite базу података која ће бити у унутар нашег процеса, тј. база података ће бити интерна а остали процеси(књиговодство, исл.) ће моћи само да гледају на њу, без могућности да је мењају. Ова база података ће чувати све информације о поруџбеницама и захтевима за набавку у предузећу.

6. БАЗА ПОДАТАКА ДОБАВЉАЧА

Користићемо SQLite базу података у којој чемо имати све договорене добављаче и цене за договорене производе. База ће се пунити преко посебног интерфејса која ће бити доступна само набаваци.

7. БАЗА ПОДАТАКА ЗА ПРОИЗВОДЕ

База података ће се качити за глану базу података предузећа из које ће прикупљати информације о шифрама репроматеријала и количини преосталог материјала у магацину. База података ће сигнализирати набавци када нешто од репроматеријала сиђе испод дозвољене количине

4.4 ПАКЕТ controller

```java
import com.pj.asss.frizerski_salon.entity.Salon;
import com.pj.asss.frizerski_salon.service.SalonService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;

import java.util.List;

@RestController
public class SalonRestController {

    @Autowired
    private SalonService salonService;

    @GetMapping("/saloni")
    public List<Salon> findAllSalonByTipUsluge() { return salonService.findAllByTipUsluge(); }

    @GetMapping("/saloni/pol")
    public List<Salon> findAllSalonByDescending() { return salonService.findAllSalonByDescending(); }

    @DeleteMapping("/saloni/{id}")
    public String deleteById(@PathVariable int id) { return salonService.deleteById(id); }

    @PutMapping("/saloni/{id}")
    public String updateById(@PathVariable int id, @RequestBody Salon salon) {
        return salonService.updateById(id, salon);
    }

    @PostMapping("/saloni")
    public Salon save(@RequestBody Salon salon) { return salonService.save(salon); }
}
```

У пакету **conroller** налази се класа **SalonRestController.** У овој класи се извршава мапирање функционалности Service слоја ради приступа и позивања истих поља, изван апликације.

4.5 ПАКЕТ aspect

```java
import org.springframework.stereotype.Component;

import java.io.*;

@EnableAspectJAutoProxy
@Aspect
@Component
public class logger {
    @Pointcut("execution(* com.pj.asss.frizerski_salon.controller.*.*(..))")
    public void allRESTMethods() {
    }

    @Around("allRESTMethods()")
    public Object arroundAllMethods(ProceedingJoinPoint joinPoint) {
        String methodName = joinPoint.getSignature().getName();
        String before = "Izvrsavanje " + methodName + " metode";

        writeLog(before);
        Object retVal = null;

        try {
            retVal = joinPoint.proceed();
            String success = "\nMetoda " + methodName + " je uspesno izvrsena.";
            writeLog(success);
        } catch (Throwable throwable) {
            String failed = "\nMetoda " + methodName + " nije uspela. " + throwable.getMessage() + " .***";
            writeLog(failed);
        }
        String after = "\n\n";
        writeLog(after);
        return retVal;
    }
```



```java
public void writeLog(String logger) {
    File file = new File( pathname: "izvestaj.txt");
    try (Writer writer = new FileWriter(file,  append: true)) {
        writer.write(logger);
    } catch (FileNotFoundException fileNotFoundException) {
        fileNotFoundException.printStackTrace();
    } catch (IOException ioException) {
        ioException.printStackTrace();
    }
}
```

У **aspect** пакету смештена је класа **logger**, у којој се налази процес креирања једне текстуалне дадотеке **izvestaj.txt**, која уписује који су све методи извршени и да ли су извршени, уколико метод није извршен,

избацује се Exception, и порука да је метод неуспесно извршен.

5.0 ЗАКЉУЧАК

Spring Framework је радни оквир за платформу Java који обезбеђује инфраструктуру за развој апликације.

Spring Boot је Спрингово решење за конфигурацију преко конвенције за креирање самосталних апликација