**КУРСОВ ПРОЕКТ**

**по Управление на софтуерни проекти**

**Изготвили:**

**Тодор Пенчев**, №20621541

**Момчил Милков**, №20621535

Специалност СИТ, III курс, 2a група

Съдържание

[Техническо задание на проекта 3](#_Toc133398913)

[Предмет на техническото задание 3](#_Toc133398914)

[Изисквания към програмния продукт 3](#_Toc133398915)

[Обхват на разработката 3](#_Toc133398916)

[Бюджет 3](#_Toc133398917)

[Обосновка на избор за система на консрол на версиите 3](#_Toc133398918)

[Причини за избор 4](#_Toc133398919)

[Трудности при използването 4](#_Toc133398920)

[Настройки, нужни за работа с него 4](#_Toc133398921)

[Съпоставка на избрания софтуер с други 4](#_Toc133398922)

[Обосновка на избора за система за управление на проекти 4](#_Toc133398923)

[причина за избора 4](#_Toc133398924)

[трудности при използването 4](#_Toc133398925)

[настройки, нужни за работата с него 4](#_Toc133398926)

[мобилна/desktop/web верси 4](#_Toc133398927)

[използвани функционалности 4](#_Toc133398928)

[Обосновка и разяснения към избраната методология за разработка на софтуера 5](#_Toc133398929)

[причина за избора 5](#_Toc133398930)

[предимства на подхода 5](#_Toc133398931)

[недостатъци на подхода, открити при изпълнението на задачат 5](#_Toc133398932)

[съпоставка на подхода с други 5](#_Toc133398933)

[Описание на стъпките по реализацията 5](#_Toc133398934)

[определяне на роли 5](#_Toc133398935)

[разделяне на задачи 5](#_Toc133398936)

[определяне на срокове за: проектиране, кодиране, документиране, интегриране, тестване 5](#_Toc133398937)

[избор на програмни средства за реализацията – програмен език, IDE 5](#_Toc133398938)

[Timeline 5](#_Toc133398939)

[Изводи 5](#_Toc133398940)

[Отчети, генерирани от софтуера за управление на проекти, и анализ на отчетите 5](#_Toc133398941)

[Слаби моменти/трудности в изпълнението на задачата 5](#_Toc133398942)

# Техническо задание на проекта

## Предмет на техническото задание

Да се разработи информационна система, предоставяща услуга склад. Програмата съхранява и обработва данни за складови помещения. Системата поддържа два вида потребители администратор и оператори (складов агент) с различни роли за достъп до функционалностите в системата.

## Изисквания към програмния продукт

Операции за работа с потребители включва създаване на складови оператори от администратор, създаване на доставчици, създаване на клиенти, създаване на каса (парична наличност).

Системата поддържа операции за работа със следните събития: създаване на номенклатури, работа с фактури, приемане на стока от доставчик на доставна цена, изписване на стока на продажна цена, наблюдение за наличност на стоки в склада, наблюдение за наличност на пари в касата

Системата поддържа справки по произволен период за: доставки и доставчици, изписване и клиенти, дейност на складовите оператори, за наличности в склада, разходи, приходи, печалба, движение на наличността в касата.

Системата поддържа известия за следните събития: критичен минимум и липса на стока, критичен минимум и липса на парична наличност

## Обхват на разработката

Приложението трябва да покрива всички изисквания, които възложителят е възложил, описани в предходната точка „изисквания към програминия продукт“. Обхватът на разработка е огрничен до система обслужваща само един склас с една каса.

## Бюджет

Проектът ще бъде анализиран, проектиран и разработен от двама програмисти в срок от 3 работни седмици.

# Обосновка на избор за система на консрол на версиите

## Причини за избор

Използваме Git, като най-популярна система от този вид. Повечето хора в тази сфера на работа са добре запознати с нея и принципите й на работа.

## Трудности при използването

Тя предоставя всички необходими възможности за работа в екип по едно и също време, като всеки има собстено хранилище, върху което работи. При получаване на конфликти за някой файл трябва да се отдели време и ръчно да се разрешат.

## Настройки, нужни за работа с него

Използвайки Github Desktop приложение и нашите акаунти в Github, качването на промени по проектът беше лесно и приятно, а настойките почти никакви.

## Съпоставка на избрания софтуер с други

Използването на Git система за контрол на версиите направи следнето на промени в ретроспекция сравнително лесно, както и кой от разработчиците е направил съответната промяна. Към всеки commit има заглавие и описание, което помага при търсене на промяна. За разлика от други системи, тук хранилището е централизирано, а всеки от разработчиците работи с локално копие на проекта. Това дава възможност няколко човека да работят по един и същ файл в даден момент от времето без да си пречат.

# Обосновка на избора за система за управление на проекти

## причина за избора

## трудности при използването

## настройки, нужни за работата с него

## мобилна/desktop/web верси

## използвани функционалности

# Обосновка и разяснения към избраната методология за разработка на софтуера

## Причина за Избора

Повечето въпроси свързани с методологията на разработката произлизат от факта, че нашият екип е съставен само от двама души. Това означава, че всеки един от нас има изключително голям контрол над крйната версия на проекта, тоест всеки един от нас прави голям брой промени и взима голям брой решения, на практитка през целия процес на разработка. Scrum методологията, адаптирана към нашите нужди, най- точно пасва на нашата задача и възможности. Изключително важен за нас фактор при правенето на този избор е гъвкавостта и свободата за итериране и оптимизиране на по- ранен код, от гледната точка на по- нататъчен етап от проекта.

## Предимства на Подхода

Както е по-горе споменато, за нас най- голямото предимство на подхода е свободата и гъкъвостта, които той предоставя. Адаптирайки спринтовете ни да траят по 1-2 сецмици, като правим междинни срещи и разговори за това какво можем да променим, какво ни харесва и как ще продължаваме се оказва изключително ефективен и бърз начин на работа в контекста на малкия ни екип. Писането и пренаписването на код става изключително бързо, понеже няма никакво административно забавяне свързано с одобряване от останалите участници, което би съществувало в по- голям екип. За всеки един такъв спринт имаме определени задачи които сме разпределили и сме решили, че трябва да свършим, но това не ни ограничава от това, да отделим част от времето си над друга част от проекта, ако някой има нова идея. Много помага факта, че имаме постоянна 24/7 връзка за кореспонденция, така че всякакви моментни въпроси могат да се решат за броени минути.

## Недостатъци на Подхода, Открити при Изпълнението на Задачата

В контекста на нашата задача тази методогия действа изключително ефективно, така че не сме попадали на особено проблемни недостатъци, или затруднения. Единственото което може да се определи в тази категория, би било, че с голямата свобода присъства един малък фактор на хаос и не- пълна яснота на това как ще изглежда крайната картинка, което може да причини някакво забавяне, но тези моменти са от излключително маловажно и нерелевантно за процеса на работа естество.

## Съпоставка на Подхода с Други

Определено, всеки един от по- статичните и предваритено планирани подходи като Waterfall и постъпковия подход не биха паснали на нашата задача, понеже биха били прекалено ограничаващи. Hacking методологията също не би била много подходяща, понеже въпреки че е по гъвкава, за нас постоянната кореспонденция и планиране на следващите етапи е много важно. Импровизирането и движението напред без план не е ефективна дългосрочна стратегия. Методи като Lean или XP (екстремно програмиране), биха имали известен успех, но основен недостатък при тях за нас е акцента върху максимизирането на ефикасността на използването на ресурсите и времето ресурсите, което е нещо което не е от толкова голямо значение за нас, понеже можем да си позволим забавяния и преизмисляния в определени моменти. Най- близкият до нашия метод би бил Kanban, като с него бихме постигнали почи идентични резултати, поради големите му прилики със Scrum.

# Описание на Стъпките по Реализацията

## Определяне на Роли

Поради малкият екип, в проекта липсват строго определени роли между участниците. Всеки един има задължението да участва в реализацията на всеки аспект на системата грубо казано разделено 50/50.

## Разделяне на Задачи

Планиранто, проектирането и документирането са общи дейности, извършени и от двамата участници. Основното разделение на задачите е при реализацията (кодирането). Проектът е грубо разделен на 4 модула: ORM + база данни, Бизнес Логика, Интерфейс (Front End), Логване. За ORM и база данни се грижи Момчил, за Front End и логване Тодор, а за бизнес логиката, и двамата, респективно занимавайки се с частите които се отнасят към по- рано споменатите им задачи.

## Определяне на Срокове за: Проектиране, Кодиране, Документиране, Интегриране, Тестване + Timeline

Както по- рано беше споменато, работата се върши на базата на 1-2 седмични спринтове, като повечето от задачите във всеки един от тях траят 1-3 дни. Кратък обзор на графика:   
-> Начално планиране на проекта (1 ден)   
-> Разработка на БД + ORM (2 дни)   
-> Проектиране на бизнес логика (5 дни)   
-> Разработка на интерфейс (3 дни)   
-> Разработка на модул за логване (1 ден)   
-> Тестване (1 ден)   
-> Докуметиране (1 ден).  
За пълния списък, включващ всички задачи, тяхното времетраене и предвидени дати на започване и завършване, можете да се обърнете към mpp файла, в папката с документацията.

## Избор на Програмни Средства за Реализацията – Програмен Език, IDE

Реализацията на проекта се извършва на няколко етапа, като първият от тях е изграждането на база данни. Това става чрез инструментите за БД на Oracle- Data Modeler и SQL Developer. Основния програмен език на проекта е Java 8, като за среда се използва IntelliJ. За ORM слоя (връзката на Java приложението с базата данни) се използва Java библиотеката Hibernate. За проектирането на Front End интерфейса използваме инструмента Scene Builder, а за свързването му с Java пиложението разчитаме на библиотека JavaFX. За logging модула използваме библиотеката log4J. Финално, за провеждането на тестове се ползва framework-а за тестване Junit. За изчерпателна информация по реализацията, можете да се обърнете към техническата документация, в папката с документацията.

# Изводи

## Слаби моменти/трудности в изпълнението на задачата

Не присъстват особено слаби моменти в изпълнението на задачата, поради качественото планиране и проектиране на проекта. Естествено, не всеки проблем е решен с първата му итерация и промените направени над предишни модули в последствие не са прекалено малко, но общо казано прогресът на работа е консистентен. Може би най- голямото затруднение би било запознаването с досега за нас непознатите технологии, нужни за реализацията на проекта (hibernate, javafx, scenebuilder, log4j, junit). Не малко количество време от създаването на приложението е посветено на запознаване и задобряване с въпросните технологии. В началото на работа с всяка една от тях, има известно забавяне, докато напълно бъдат осъзнати малките детайли и ключови моменти, нужни за използването ѝ, но след това нещата се движат сравнително безпроблемно.