

Министерство науки и высшего образования РФ  
Федеральное государственное образовательное учреждение  
высшего образования  
**«УФИМСКИЙ ГОСУДАРСТВЕННЫЙ АВИАЦИОННЫЙ ТЕХНИЧЕСКИЙ  
УНИВЕРСИТЕТ»**

Кафедра автоматизированных систем управления

Направление подготовки  
09.03.03 Прикладная информатика

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**  
к курсовой работе по дисциплине «Информационные системы»

«Разработка кроссплатформенного программного продукта на языке JAVA с  
использованием системы контроля версий»

Выполнили:  
Ст. гр. ПИ-221  
Газин Д.Р.  
Гибадуллина Э.Ю.  
Катасонов С.А.  
Рафиков Д.Р.

Проверил:  
Преподаватель  
Казанцев А.В.

Уфа – 2021

# **Министерство науки и высшего образования РФ**

Федеральное государственное бюджетное образовательное учреждение высшего образования  
«УФИМСКИЙ ГОСУДАРСТВЕННЫЙ АВИАЦИОННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра автоматизированных систем управления

## **ЗАДАНИЕ**

на курсовую работу по дисциплине «Информационные системы»

Студент Газин Д. Р. Группа ПИ-221 Консультант Казанцев А.В.  
Фамилия И.О. номер группы Фамилия И.О.

Студент Гибадуллина Э.Ю. Группа ПИ-221 Консультант Казанцев А.В.  
Фамилия И.О. номер группы Фамилия И.О.

Студент Катасонов С.А. Группа ПИ-221 Консультант Казанцев А.В.  
Фамилия И.О. номер группы Фамилия И.О.

Студент Рафиков Д.Р. Группа ПИ-221 Консультант Казанцев А.В.  
Фамилия И.О. номер группы Фамилия И.О.

1. Тема курсового проекта: Разработка кроссплатформенного программного продукта на языке JAVA с использованием системы контроля версий.  
наименование темы

2. Основное содержание:

1. Пояснительная записка с необходимыми материалами.
2. Репозиторий системы контроля версий содержащий программный код с комментариями и необходимую документацию.

3. Требования к оформлению:

3.1. Пояснительная записка должна быть оформлена в текстовом процессоре LibreOffice Writer в соответствии с требованиями СТО УГАТУ. Минимальные требования к оформлению: размер шрифта 14 пунктов; отступы от края листа: отступ слева 2 см. и остальные отступы 0.5 см. В бумажном виде оформляются: титульный лист, задание, календарный план и аннотация, которая содержит ссылку на репозиторий с программным кодом и документацией.

3.2. В пояснительной записке должны содержаться следующие разделы:

Раздел 1. Описание предметной области.

Раздел 2. Техническое задание на создание программного продукта.

Раздел 3. Настройка среды разработки для операционных систем семейств Windows и Linux.

Раздел 4. Настройка среды разработки для подключения к системе контроля версий.

Раздел 5. Реализация исходного кода по зонам ответственности.

Раздел 6. Сборка и тестирование программного продукта.

Раздел 7. Настройка программной среды для развертывания и запуска программного продукта.

Раздел 8. Руководство пользователя программного продукта.

3.3. В приложение выносится программный код и код тестов.

4. Графическая часть должна включать:

- мнемосхема рассматриваемого процесса;
- диаграммы UML;
- экранные формы инструментальных средств;
- экранные формы, разрабатываемого программного продукта.

Дата выдачи 6 марта 2021 г.

Дата окончания 29 мая 2021 г.

Руководитель \_\_\_\_\_ Казанцев А.В.

ФИО \_\_\_\_\_

## **АННОТАЦИЯ**

В ходе выполнения данного курсового проекта на тему «Разработка кроссплатформенного программного продукта на языке JAVA с использованием системы контроля версий» происходит разработка программного продукта «Калькулятор сдельно-премиальной зарплаты».

Данный калькулятор разрабатывается для предприятия ООО «Авиа-строй».

Назначение программного продукта: программа предназначена для автоматизации расчета заработной платы для сотрудника-пользователя и возможность изменения настроек для администратора.

В рамках курсовой работы происходит поэтапно создание веб-приложения: описание предметной области, моделирование системы, настройка сред разработки, написание исходного кода, тестирование компонентов программы, сборка и тестирование программного продукта и далее развертывание.

Ссылка на репозиторий с проектом и необходимой документацией:  
[https://github.com/Gibadullina/calculator-team1.](https://github.com/Gibadullina/calculator-team1)

## СОДЕРЖАНИЕ

Раздел 1. Описание предметной области.....	5
Раздел 2. Техническое задание на создание программного продукта..	13
Раздел 3. Настройка среды разработки для операционных систем семейств Windows и Linux.....	14
3.1. Настройка среды разработки для Windows 10.....	14
3.2. Linux Ubuntu 20.04.....	26
3.3. Linux Mint.....	34
Раздел 4. Настройка среды разработки для подключения к системе контроля версий.....	42
Раздел 5. Реализация исходного кода по зонам ответственности.....	50
Раздел 6. Сборка и тестирование программного продукта.....	52
Раздел 7. Настройка программной среды для развертывания и запуска программного продукта.....	59
Раздел 8. Руководство пользователя программного продукта.....	62
Раздел 9. Руководство программиста программного продукта.....	63
Заключение.....	64
Приложения.....	65
Список литературы.....	168

## **Раздел 1. Описание предметной области**

Темой данной курсовой работы является создание зарплатного калькулятора для сдельной формы оплаты труда. Поэтому будут рассмотрены такие понятия, как: оплата труда (заработка плата), формы оплаты труда, сдельная оплата труда и ее типы.

Заработка плата (оплата труда работника) - вознаграждение за труд в зависимости от квалификации работника, сложности, количества, качества и условий выполняемой работы, а также компенсационные выплаты (доплаты и надбавки компенсационного характера, в том числе за работу в условиях, отклоняющихся от нормальных, работу в особых климатических условиях и на территориях, подвергшихся радиоактивному загрязнению, и иные выплаты компенсационного характера) и стимулирующие выплаты (доплаты и надбавки стимулирующего характера, премии и иные поощрительные выплаты).

Формой оплаты труда можно назвать механизм начисления заработной платы, который обеспечивает учет количества труда, затраченного работниками. Выделяют 2 основных формы оплаты труда:

- Повременная: оплата труда производится за фактически отработанное время, независимо от результатов работы;
- Сдельная: оплата труда производится за объем выполненных работ, независимо от потраченного времени.

На практике выделяют следующие виды сдельной оплаты труда:

- Сдельно-премиальная оплата труда;
- Прямая сдельная оплата труда;
- Сдельно-прогрессивная оплата труда
- Аккордная оплата труда;
- Смешанная (повременно-сдельная) оплата труда.

Сдельно-премиальная система оплаты труда, наряду с оплатой по прямым сдельным расценкам, предусматривает премирование за перевыполнение нормы выработки и за достижение количественных и качественных показателей, определенных действующими условиями премирования.

По прямой сдельной системе заработка плата начисляется исходя из объема выполненной работы с использованием твердых сдельных расценок.

Оплата труда при сдельно-прогрессивной системе в пределах установленных норм производится по прямым сдельным расценкам, а сверх этих норм - по повышенным расценкам.

При аккордной системе оплаты труда заработка устанавливается на весь объём работы, а не на отдельную операцию. При этом устанавливается предельный срок выполнения работы.

Смешанная оплата труда представляет собой синтез сдельной и повременной оплаты труда.

В данной курсовой работе будет рассматриваться отдельно сдельно-премиальная форма оплаты труда. Сдельно-премиальная оплата труда предусматривает оплаты по прямым сдельным расценкам и премирование за перевыполнение нормы. Следовательно, важно учесть следующие понятия: сдельные расценки, премия.

Сдельные расценки - это размер заработной платы за единицу выполненной работы или изготовленной продукции.

Премия - это один из видов поощрения работника, добросовестно исполняющего трудовые обязанности, размер и условия выплаты которого работодатель определяет с учетом совокупности обстоятельств, предусматривающих самостоятельную оценку им выполненных работником трудовых обязанностей, и иных условий, влияющих на размер премии, включая результаты экономической деятельности самой организации.

Также учтем районный коэффициент, МРОТ для дальнейших расчетов.

Районный коэффициент — выплаты, обусловленные районным регулированием труда, предназначенные для компенсации повышенных физиологических нагрузок в связи с работой в неблагоприятных климатических условиях.

МРОТ (минимальный размер оплаты труда) - установленный минимум оплаты труда.

Документы, регламентирующие данную деятельность:

- 1) Федеральный закон от 19.06.2000 № 82-ФЗ (ред. от 27.12.2019) "О минимальном размере оплаты труда";
- 2) Раздел VI ТК РФ от 30.12.2001 N 197-ФЗ (ред. 09.11.20) Оплата и нормирование труда;
- 3) Федеральный закон от 12 января 1996 г. N 10-ФЗ "О профессиональных

- союзах, их правах и гарантиях деятельности" (ред. От 8 декабря 2020 г.);
- 4) Статья 315 ТК РФ от 30.12.2001 N 197-ФЗ (ред. 09.11.20) Оплата труда;
  - 5) Статья 316 ТК РФ от 30.12.2001 N 197-ФЗ (ред. 09.11.20) Районный коэффициент к заработной плате.
  - 6) Статья 133 ТК РФ от 30.12.2001 N 197-ФЗ (ред. 09.11.20) Установление минимального размера оплаты труда.
  - 7) Глава 23 НК РФ ч.2 от 5.08.2000 N 117-ФЗ (ред. 23.11.20) Налоги на доходы физических лиц.

В курсовой работе ведется разработка калькулятора заработной платы для ООО «Авиа-строй», которое занимается производством деталей реактивных двигателей:

- компрессоры;
- вентиляторы;
- турбины;
- сопло;
- смесители.

В данном разделе курсовой работы для представления наглядного графического изображение системы или процессов системы в символьно-графической форме включена мнемоническая схема (мнемосхема) бизнес-процесса, которая визуализирует процесс работы пользователя с ПО. Мнемосхема состоит из графических элементов, взаимосвязанных между собой при помощи соединительных линий. Сама мнемосхема представлена на рисунке 1.

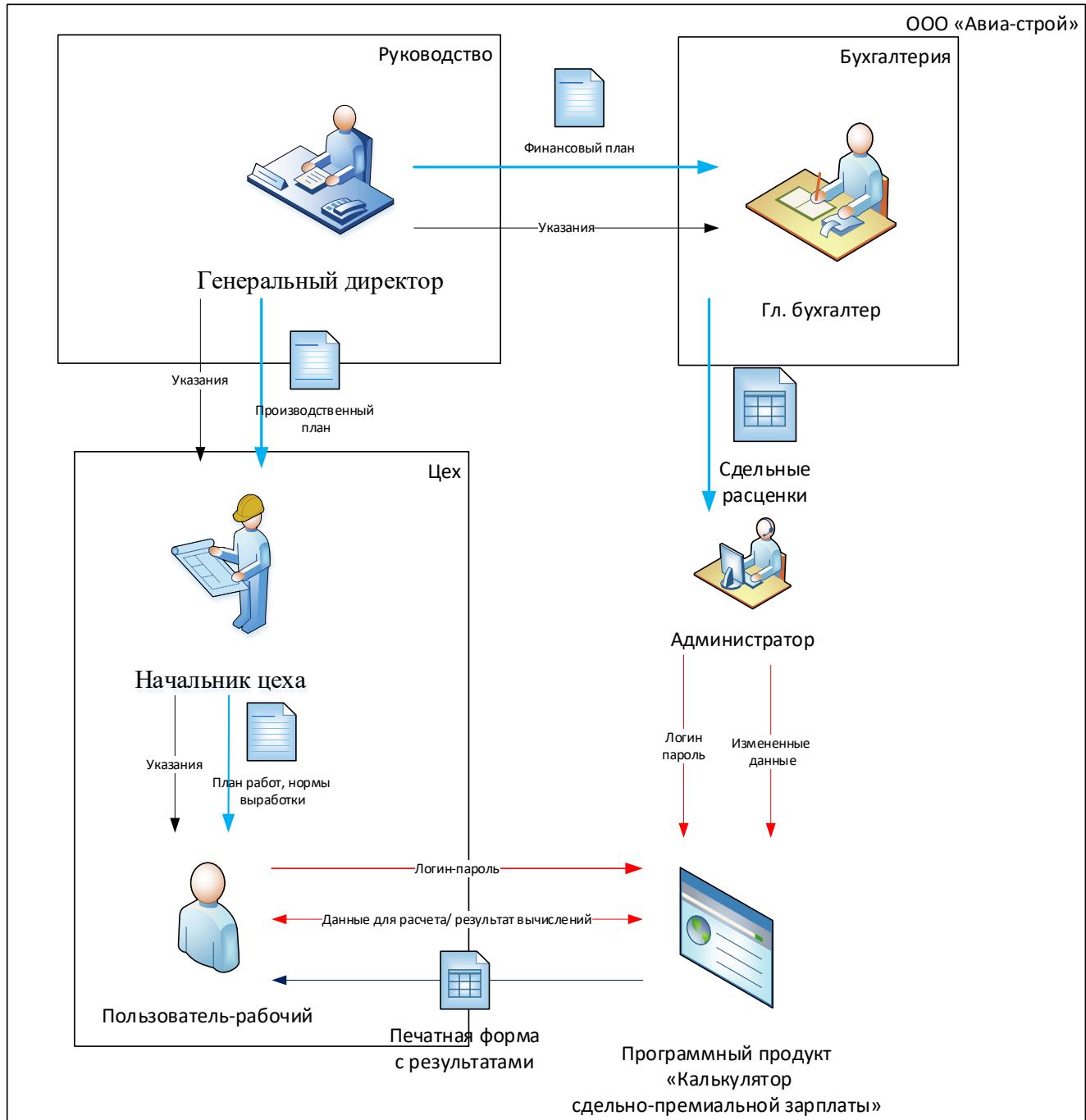


Рисунок 1 - Мнемосхема бизнес-процесса

Диаграмма вариантов использования позволяет описать систему (процесс в системе) на концептуальном уровне и демонстрирует функциональные возможности программного продукта (рис.2).

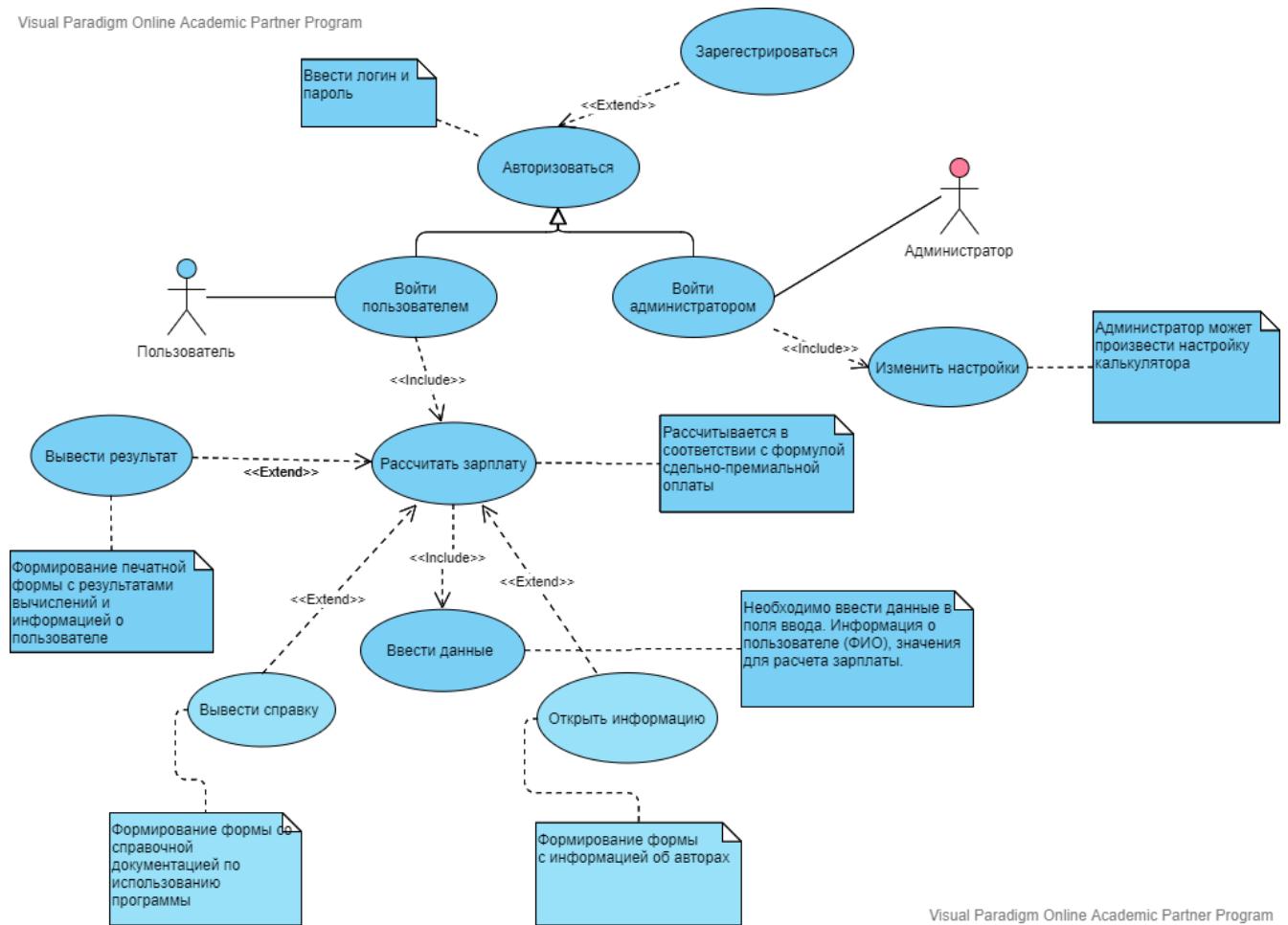


Рисунок 2 - Диаграмма вариантов использования

Диаграмма классов демонстрирует общую структуру классов (рис.3).

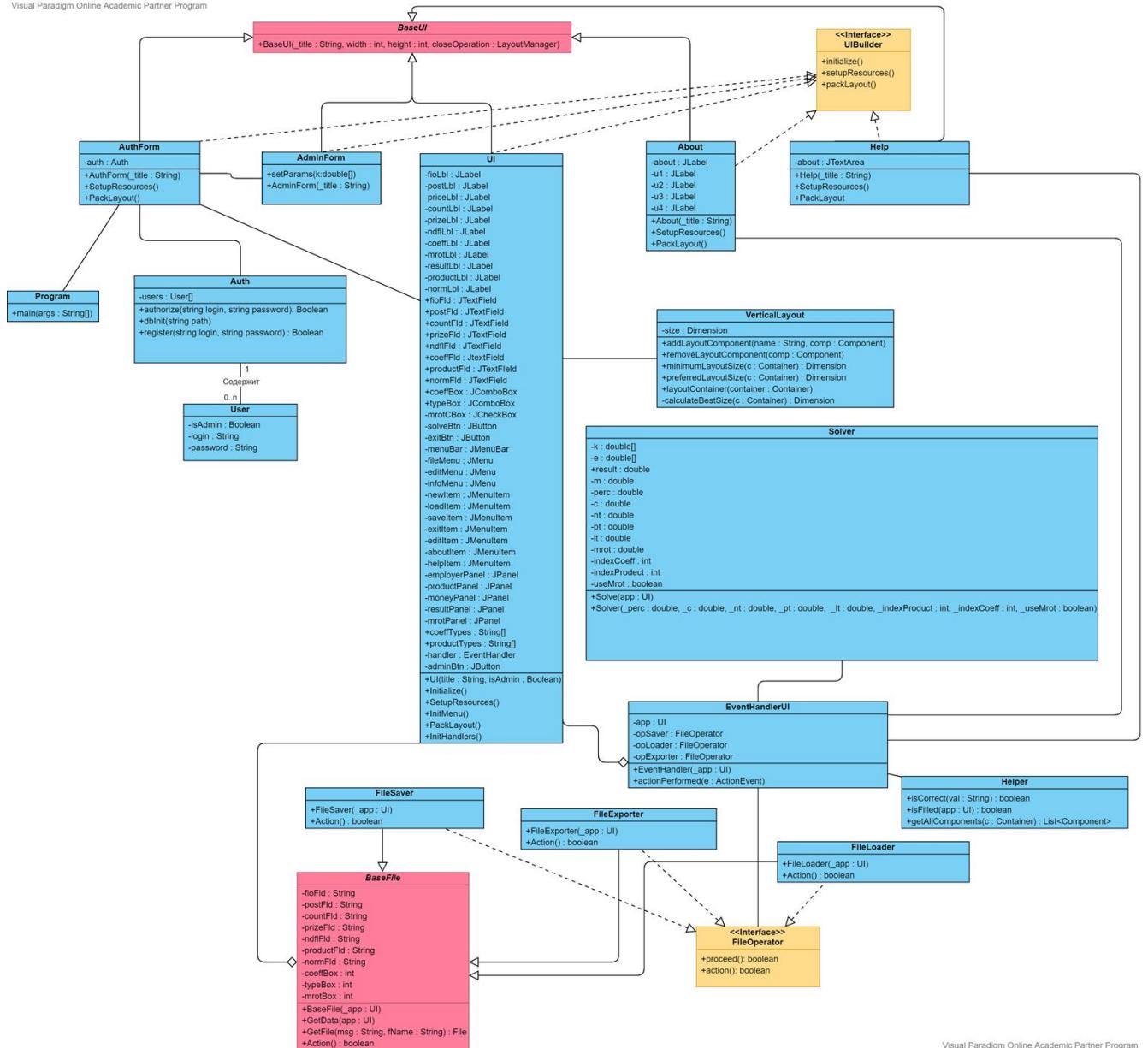


Рисунок 3 - Диаграмма классов

Описание входной информации. Входная информация — это необходимая информация для расчета заработной платы, которую вводит пользователь. Поля ввода данных, которые должен заполнить пользователь для получения результата:

- ФИО работника;
- должность работника;
- изготовленная продукция;
- премия;
- единицы товара для премии;
- НДФЛ;
- норма.

Также пользователю предоставляется выбор одного из вариантов выпадающего списка:

- расценка за единицу продукции;
- районный коэффициент.

Для администратора входная информация вводится в меню настройки. У него есть возможность:

- изменить расценку за единицу продукции — турбина;
- изменить «Республика Башкортостан» на другой регион, соответственно поменяв значение районного коэффициента.

Описание выходной информации. Выходная информация представляет собой результат расчета заработной платы, который отображается на форме. Также предусматривается формирование печатной формы *.xls*, содержащей не только результаты, но и данные пользователя.

Для администратора выходная информация — это файл настройки, в котором показаны изменения полей ввода главной экранной формы для пользователя.

Математическая модель «Калькулятора сдельно-премиальной зарплаты».

Заработка плата при сдельно-премиальной форме оплаты труда рассчитывается по формуле 1.

$$M_{общ} = M \cdot k \cdot (1 - perc \div 100), (1)$$

где  $M_{общ}$  — размер заработной платы, руб.;

$M$  — зарплата без учета налога и районного коэффициента, руб;

$perc$  — налог, %;

$k$  — районный коэффициент.

Зарплата без учета налога и районного коэффициента рассчитывается по формуле (2).

$$M = e \cdot c + (c - l) \div nt \cdot pt, (2)$$

где  $e$  — расценка за единицу продукции, руб;

$c$  — количество изготовленной продукции, ед.;

$nt$  — товары необходимые для премии, ед;

$l$  — норма произведенных товаров, ед;

$pt$  — надбавка за премиальное кол-во товара.

Если норма произведенных товаров больше, чем количество произведенных товаров, то используется формула (3).

$$M = e \cdot c, (3)$$

Заработка плата при условии, что  $M < M_{ПОТ}$  рассчитывается по формуле (4).

$$M_{общ} = M_{ПОТ} \cdot k \cdot (1 - perc \div 100), (4)$$

где МРОТ — минимальный размер оплаты труда, руб.

Следующие значения являются константами:

- сделенная расценка для компрессора — 1000 руб;
- сделенная расценка для вентилятора — 300 руб;
- сделенная расценка для турбины — 500 руб;
- сделенная расценка для сопла -100 руб;
- сделенная расценка для смесителя — 200 руб;
- МРОТ — 12130 руб.;
- районный коэффициент для Республики Башкортостан — 1,15;
- районный коэффициент для Ростовской области -1,1;
- районный коэффициент для Республики Дагестан — 1,2;
- районный коэффициент для Республики Калмыкия -1,3;
- районный коэффициент для Ставропольского края -1,15.

Пример расчета при выборе следующих параметров: Республика Башкортостан, расценка за единицу продукции — 1000 руб.; количество изготовленной продукции — 100 ед.; надбавка за премиальное кол-во товара — 10 руб; товары для премии — 10 ед.; норма — 10 ед.; налог — 13 %.

$$1) M = 1000 \cdot 100 + (100 - 10) \div 10 \cdot 10 = 100090 \text{руб.}$$

$$2) M_{\text{общ}} = 100090 \cdot 1,15 \cdot (1 - 13 \div 100) = 100140,04 \text{руб.}$$

Пример расчета, когда значение заработной платы ниже значения МРОТ.

Выбраны следующие параметры: Республика Калмыкия, расценка за единицу продукции — 100 руб.; количество изготовленной продукции — 100 ед.; надбавка за премиальное кол-во товара — 100 руб; товары для премии — 10 ед.; норма — 100 ед.; налог — 13 %, МРОТ — 12130 руб.

$$1) M = 100 \cdot 100 + (100 - 10) \div 10 \cdot 100 = 10900 \text{руб.}$$

$M < \text{МРОТ}$ , так как  $10900 < 12130$ .

$$2) M_{\text{общ}} = 12130 \cdot 1,3 \cdot (1 - 13 \div 100) = 13719,03 \text{руб.}$$

## **Раздел 2. Техническое задание на создание программного продукта**

Техническое задание (ТЗ) — исходный документ, который является основанием для разработки и испытания программы или автоматизированной системы. Техническое задание на программу и программное обеспечение разрабатывается в соответствии с требованиями ГОСТ 19.201-78.

В ходе выполнения курсовой работы для программного продукта было разработано техническое задание для программного продукта «Калькулятор сдельно-премиальной зарплаты» (см. ПРИЛОЖЕНИЕ 1).

## **Раздел 3. Настройка среды разработки для операционных систем семейств Windows и Linux**

В данном разделе содержится порядок настройки среды разработки Eclipse в операционных системах семейств Windows и Linux. При этом рассматривается один дистрибутив из семейства Windows и два различных дистрибутива из семейства Linux.

Из семейства Windows – это Windows 10, а из семейства Linux – это Ubuntu 20.04 и Mint.

### **3.1. Настройка среды разработки для Windows 10**

Сначала необходимо скачать JDK для вашей операционной системы по ссылке: <https://www.oracle.com/java/technologies/javase/jdk8-downloads.html> (рис.4)

Windows x86	154.69 MB	 jdk-8u281-windows-i586.exe
Windows x64	166.97 MB	 jdk-8u281-windows-x64.exe

Рисунок 4 - JDK

Производим установку JDK (рис.5).

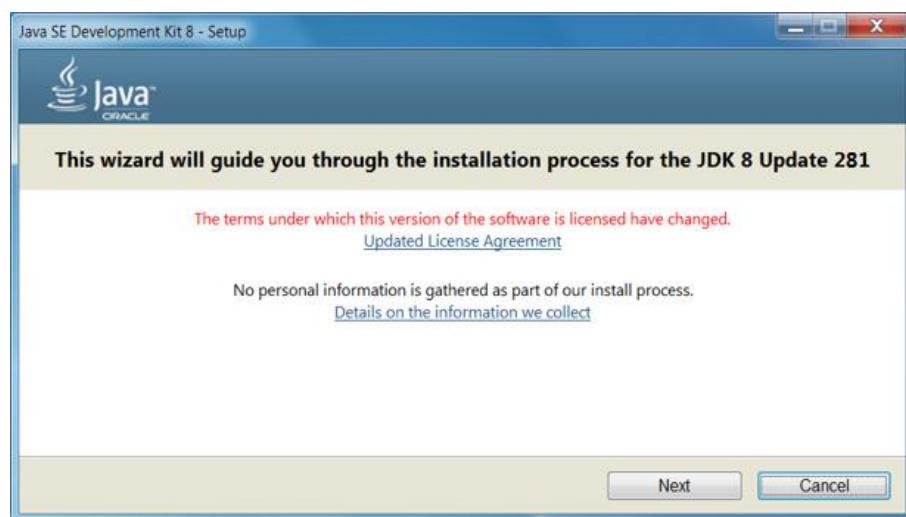
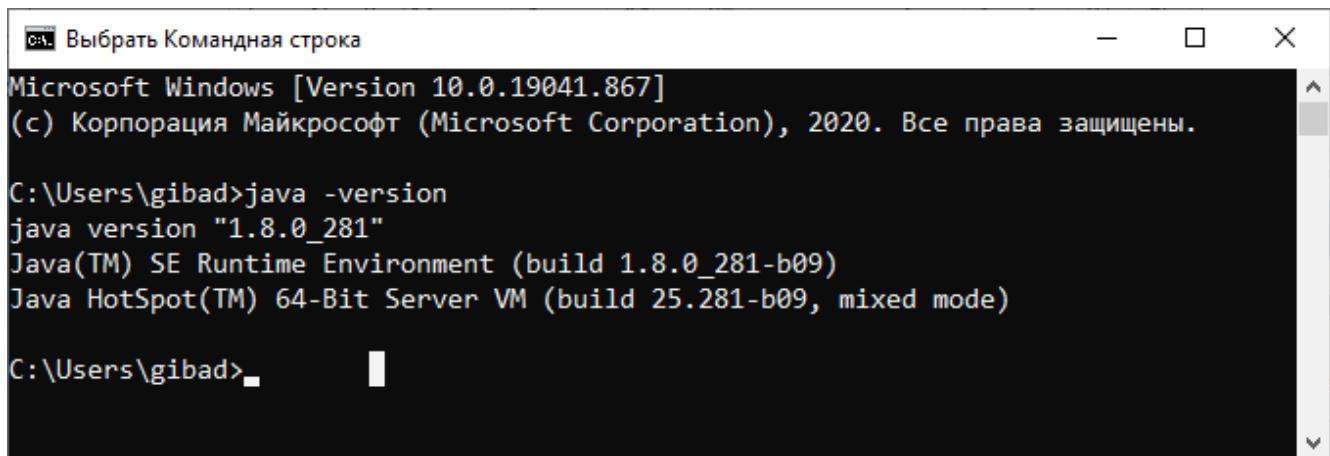


Рисунок 5 - Установка JDK

Для проверки вводим команду *java -version* (рис.6).



```
Выбрать Командная строка
Microsoft Windows [Version 10.0.19041.867]
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

C:\Users\gibad>java -version
java version "1.8.0_281"
Java(TM) SE Runtime Environment (build 1.8.0_281-b09)
Java HotSpot(TM) 64-Bit Server VM (build 25.281-b09, mixed mode)

C:\Users\gibad>
```

Рисунок 6 - Версия JDK

Далее необходимо установить среду разработки *Eclipse IDE for Enterprise Java Developers*.

- 1) Скачать Eclipse IDE для вашей операционной системы по ссылке:  
<https://www.eclipse.org/downloads/>.
- 2) Производим установку *Eclipse IDE* через *eclipseinstaller*, который вы скачали ранее (рис.7).

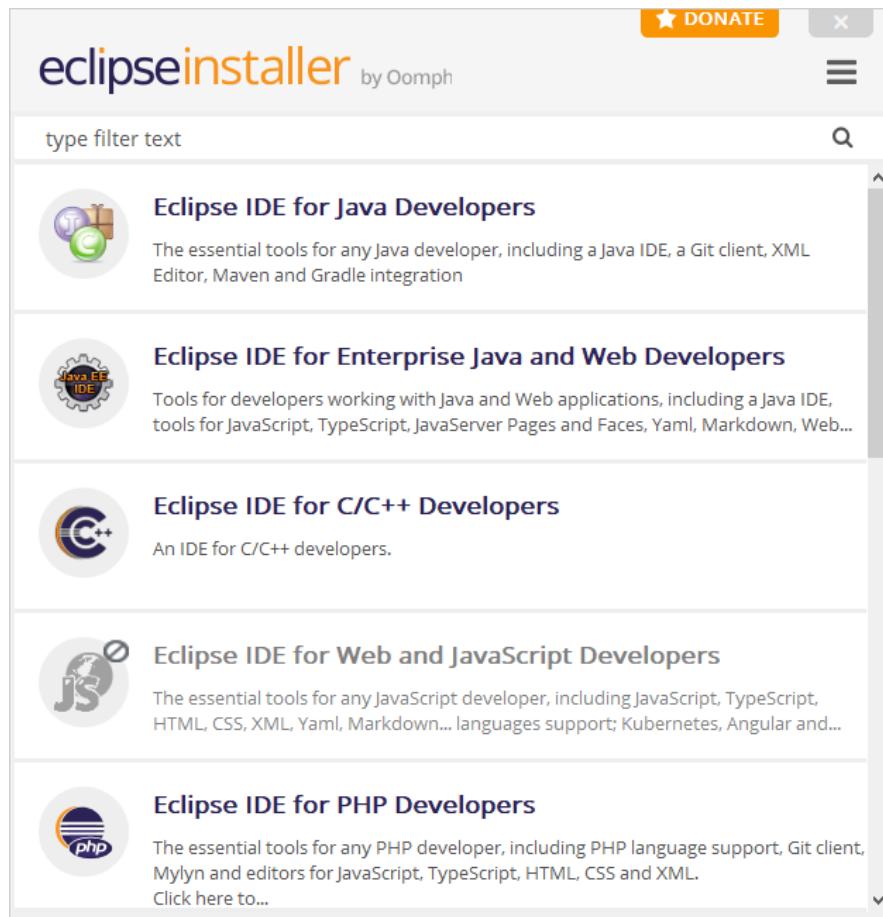


Рисунок 7 - eclipseinstaller

- 3) Выбираем *Eclipse IDE for Enterprise Java Developers* (рис.8). Указываем путь установки. Нажимаем *Install*

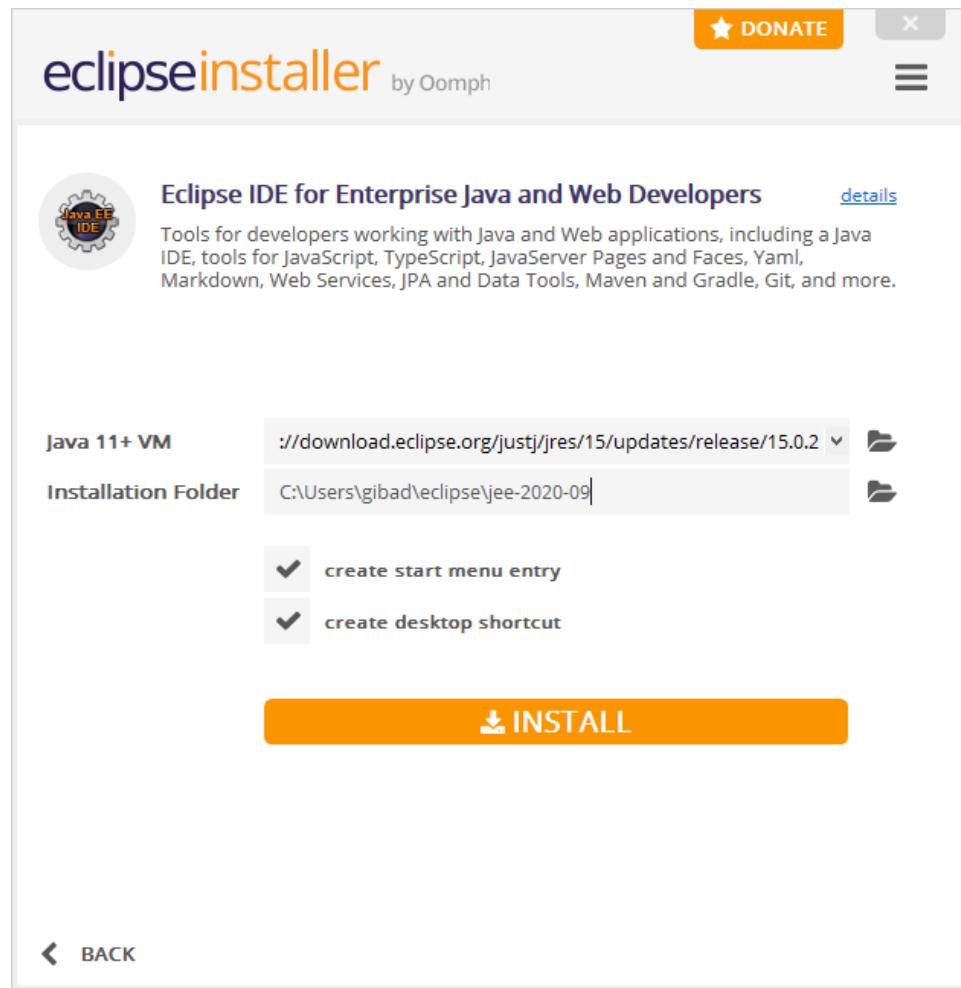


Рисунок 8 - Установка *Eclipse*

4) Окно успешной установки на рисунке 9. Далее нажимаем Launch.

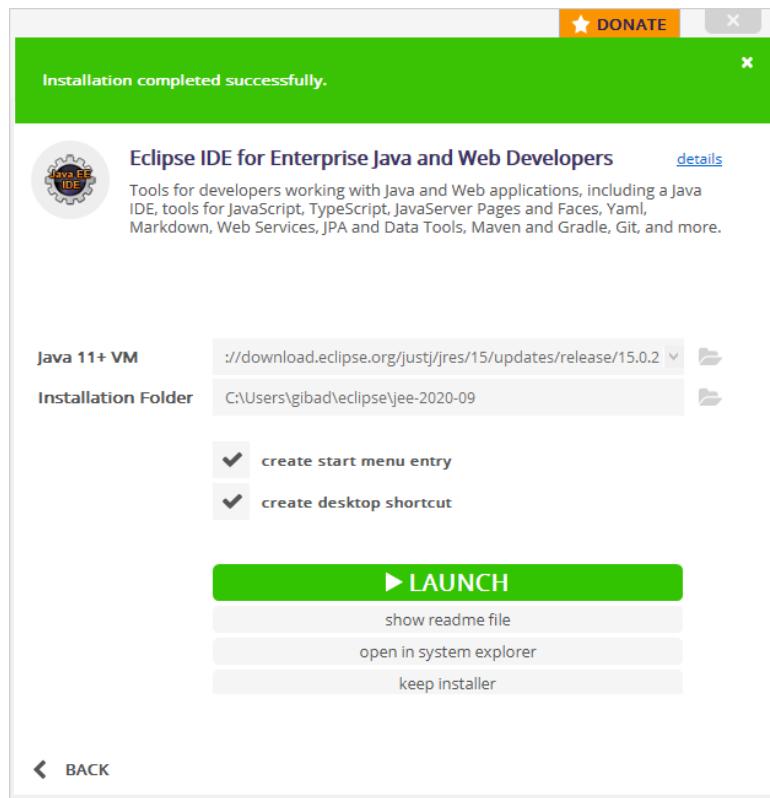


Рисунок 9 - Завершение установки

5) Указывается рабочая область (место) на рисунке 10.

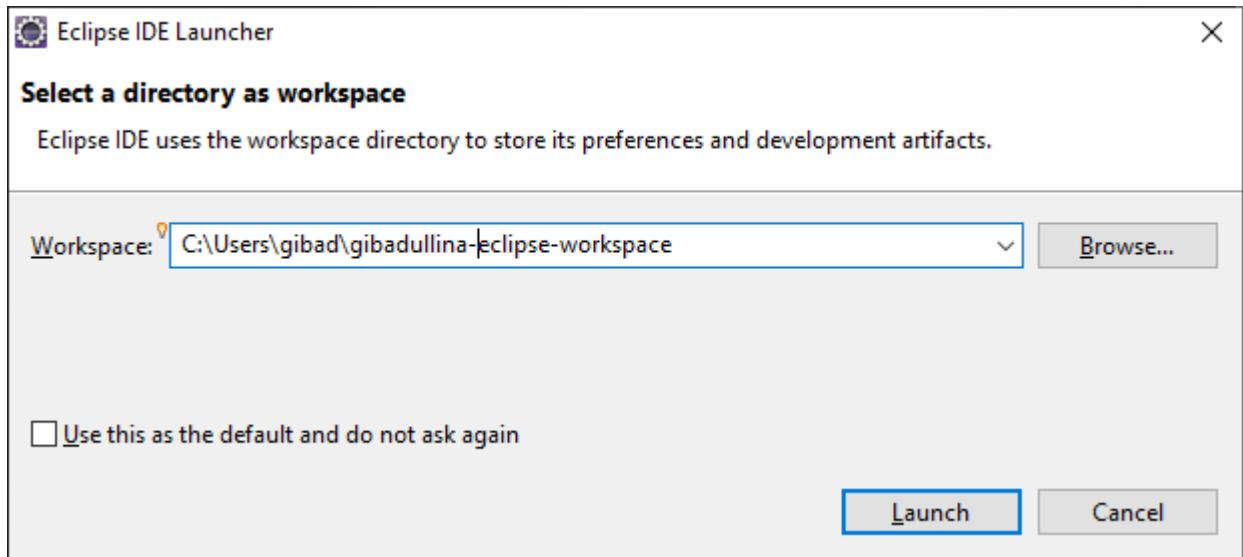


Рисунок 10 - Указание workspace

6) Запуск Eclipse (рис.11).

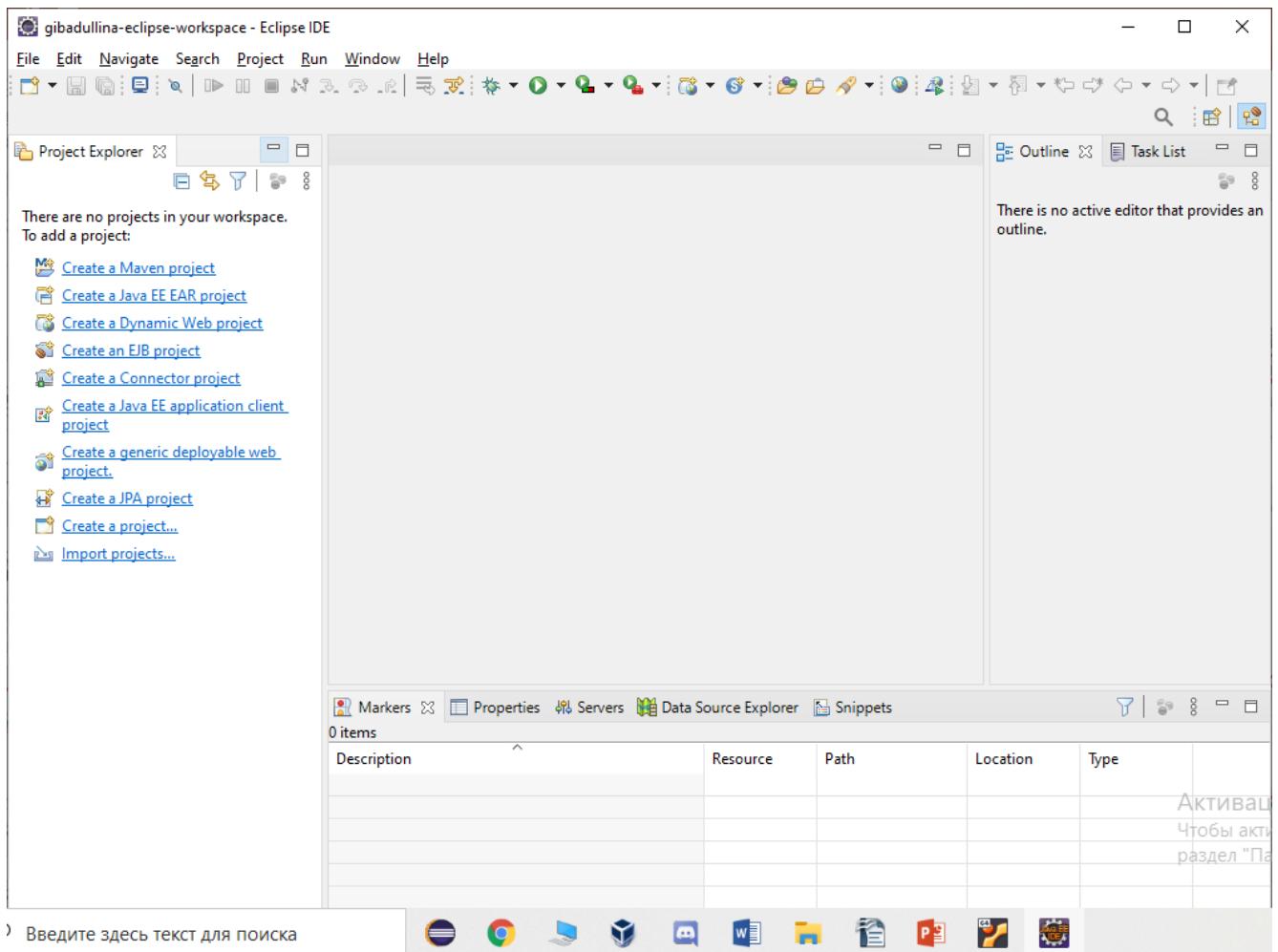


Рисунок 11 - Рабочая область Eclipse

Установка *Maven*.

Apache Maven – фреймворк для автоматизации сборки проектов на основе описания их структуры в файлах на языке POM (Project Object Model).

Установка *Maven*. Ручная установка предполагает следующие шаги:

1) Скачиваем последнюю версию *Maven* с официального сайта:

<http://maven.apache.org/download.cgi> (рис. 12)

Link	Checksums	Signature
Binary tar.gz archive	apache-maven-3.6.3-bin.tar.gz	apache-maven-3.6.3-bin.tar.gz.sha512
Binary zip archive	apache-maven-3.6.3-bin.zip	apache-maven-3.6.3-bin.zip.sha512
Source tar.gz archive	apache-maven-3.6.3-src.tar.gz	apache-maven-3.6.3-src.tar.gz.sha512
Source zip archive	apache-maven-3.6.3-src.zip	apache-maven-3.6.3-src.zip.sha512

Рисунок 12 - Версии *Maven* для скачивания

- 2) Распаковываем архив в любую директорию, например, C:\Maven (рис.13).

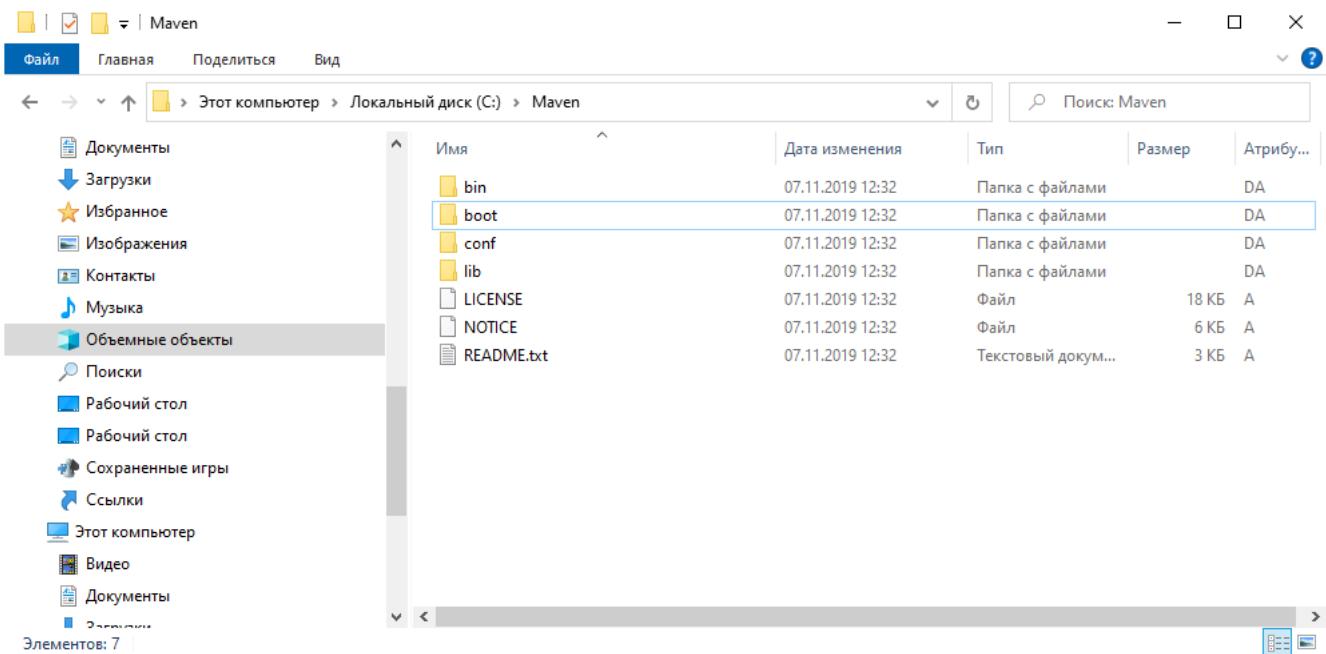


Рисунок 13 - Распаковка

- 3) Создаем переменную M2\_HOME со значением «C:\Maven»
- 4) Создаем переменную M2 со значением «%M2\_HOME%\bin».
- 5) Создаем переменную JAVA\_HOME с путем к JDK.
- 6) Изменяем переменную Path – дописать «%M2%», чтобы папка с исполняемым файлом Maven была видна из командной строки.

Результаты создания переменных среды (рис.14).

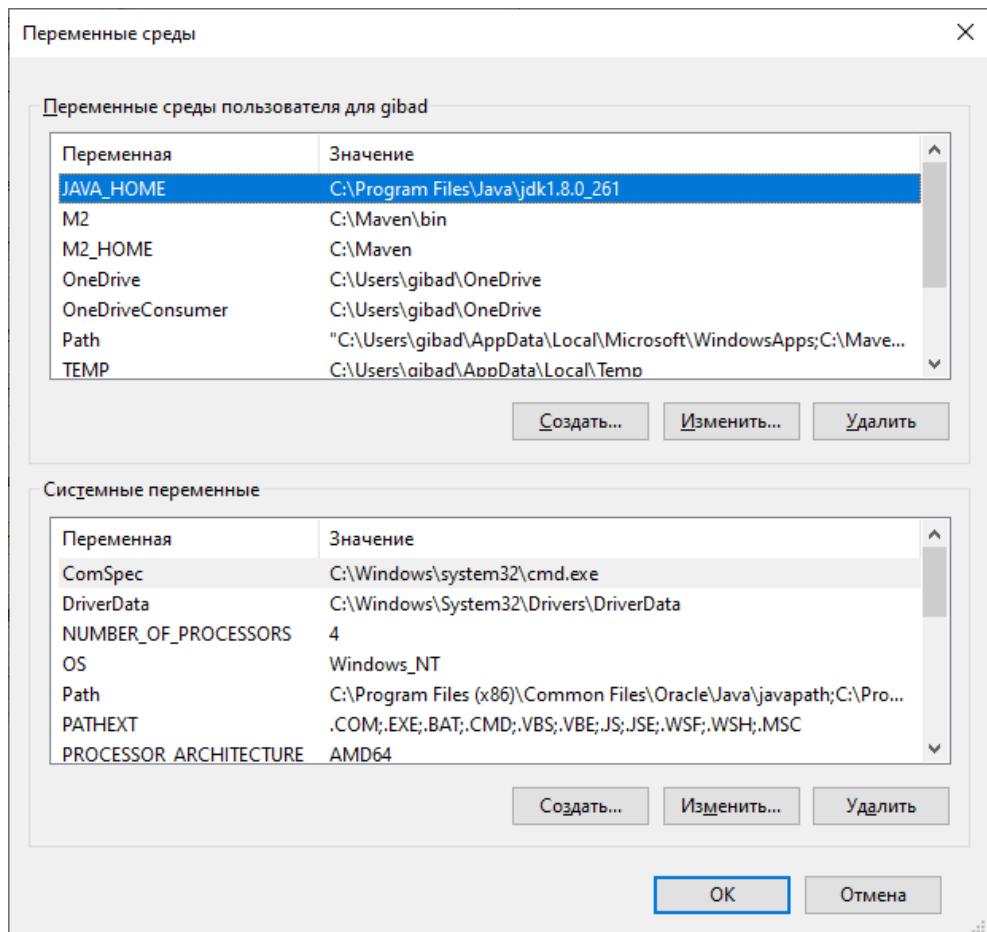


Рисунок 14- Создание переменных

7) Проверяем работу *Maven* в командной строке, где вводим команду: *mvn -version*. В итоге должна выводится информация о версиях *Maven*, *JRE* и операционной системы, как показано на рисунке 15.

```
Выбрать Командная строка
Microsoft Windows [Version 10.0.19041.867]
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

C:\Users\gibad> mvn -version
Apache Maven 3.6.3 (cecedd343002696d0abb50b32b541b8a6ba2883f)
Maven home: C:\Maven\bin..
Java version: 1.8.0_261, vendor: Oracle Corporation, runtime: C:\Program Files\Java\jdk1.8.0_261\jre
Default locale: ru_RU, platform encoding: Cp1251
OS name: "windows 10", version: "10.0", arch: "amd64", family: "windows"

C:\Users\gibad>
```

Рисунок 15- Команда mvn -version

Теперь есть возможность создания *Maven* проекта. Для этого нужно выбрать пункты File -> New -> Maven Project (рис.16)

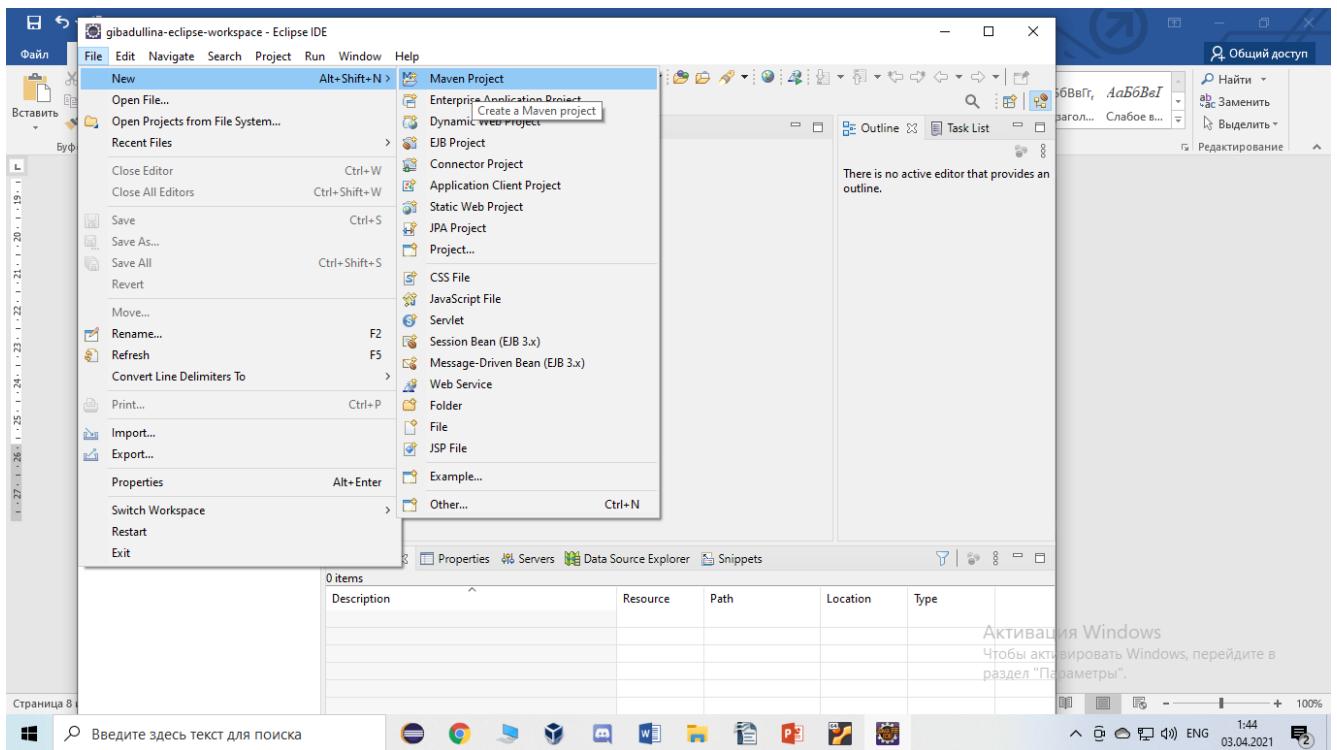


Рисунок 16 - Контекстное меню

Создаем простой проект, для этого в появившемся окне (рис.17) выбираем «Create a simple project». Нажимаем *Next*.

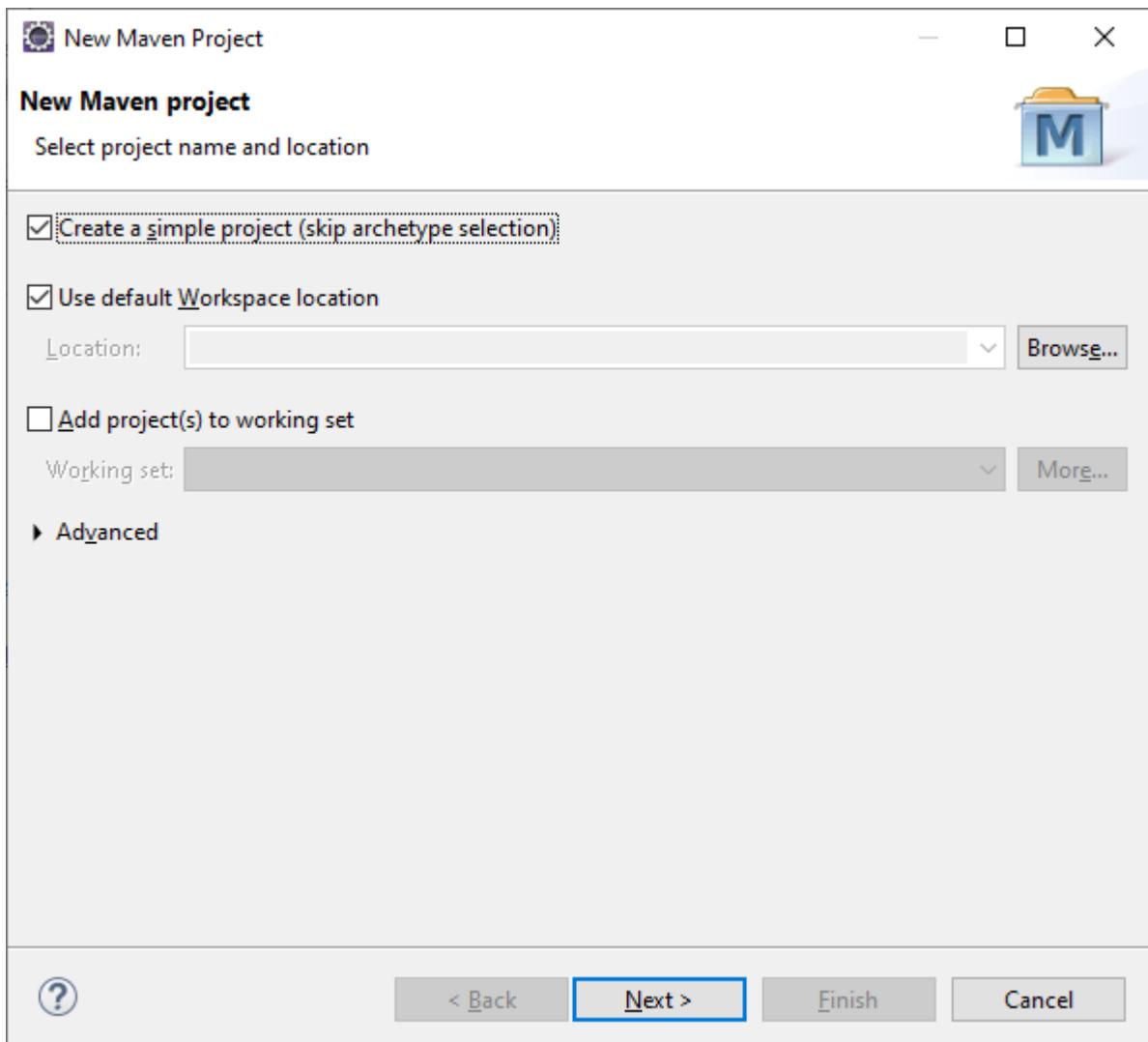


Рисунок 17 - Создание Maven project

Заполняются необходимые поля. Надо выбрать расширение WAR. Веб-приложение может быть запаковано в Web Archive (архив с расширением WAR), т.е. WAR создается вместо JAR, чтобы показать, что это веб-приложение. (рис.18)

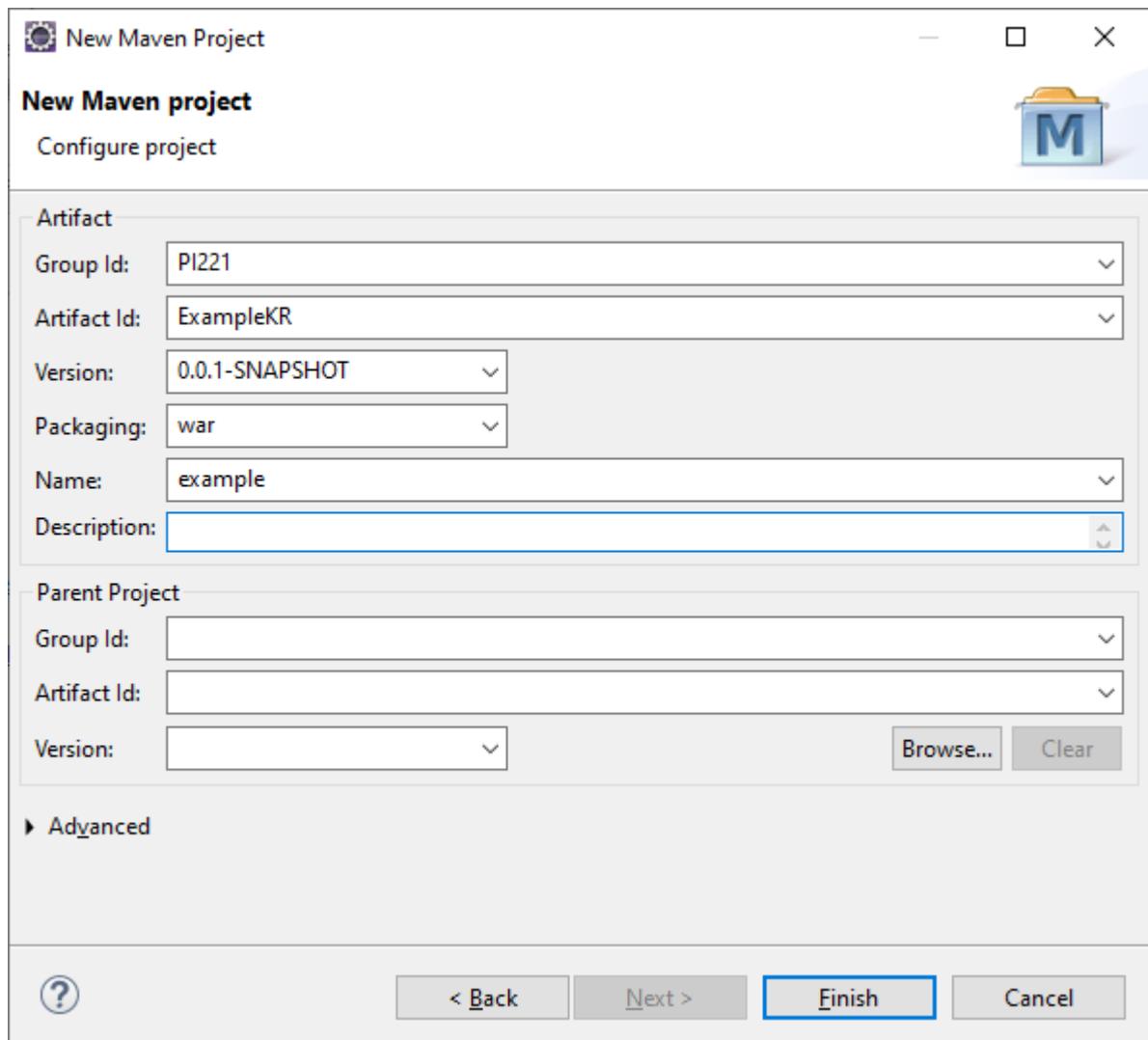


Рисунок 18 - Настройка проекта

В итоге создан Maven project (рис.19).

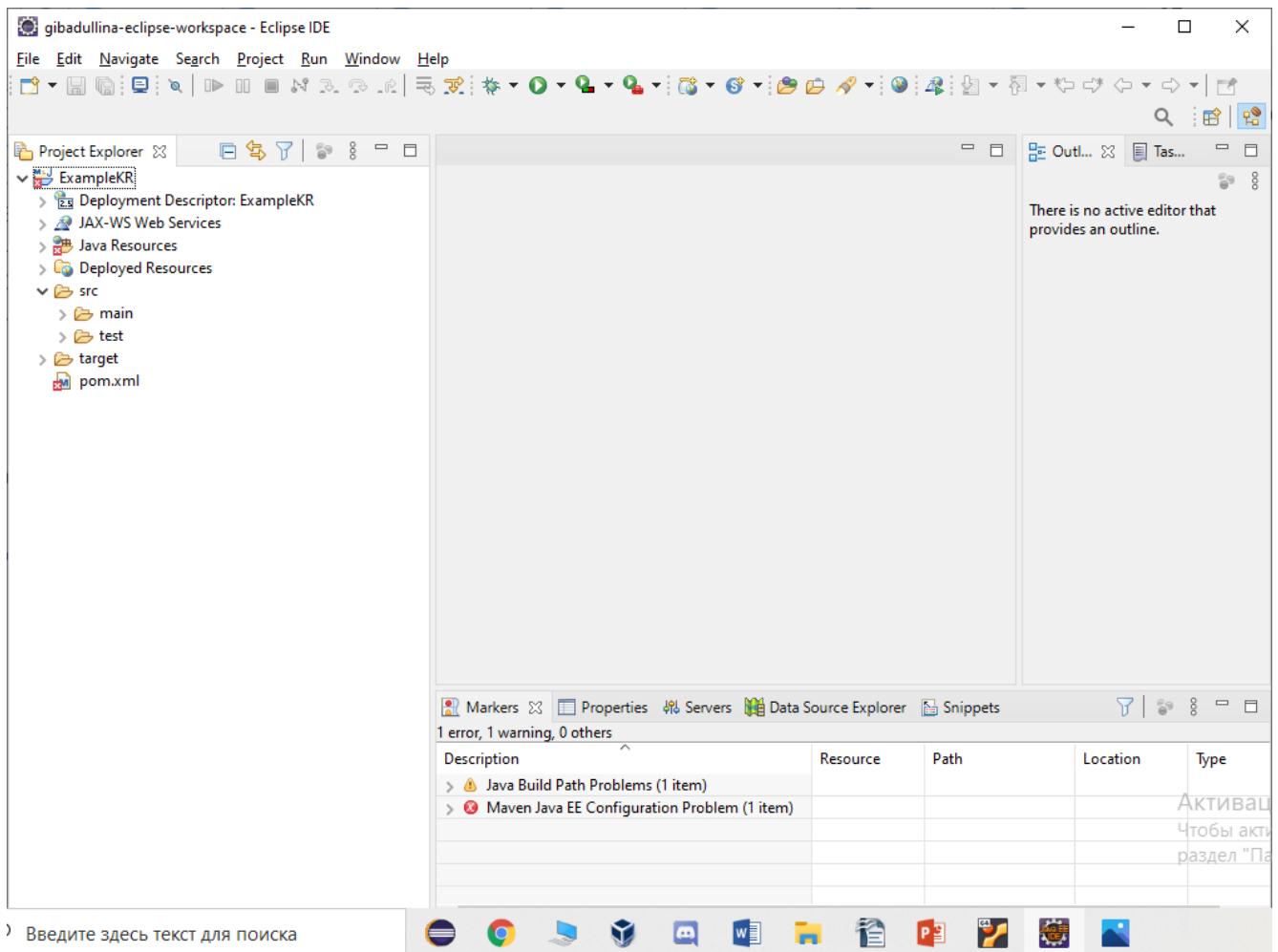


Рисунок 19 - проект ExampleKR

Настройка *Eclipse* для работы с [github.com](https://github.com). Для этого необходимы дополнительно подключаемые модули, в частности «Git». Выбираются модули «Eclipse EGit» и «Eclipse EGit Mylyn GitHub Feature» (рис.20).

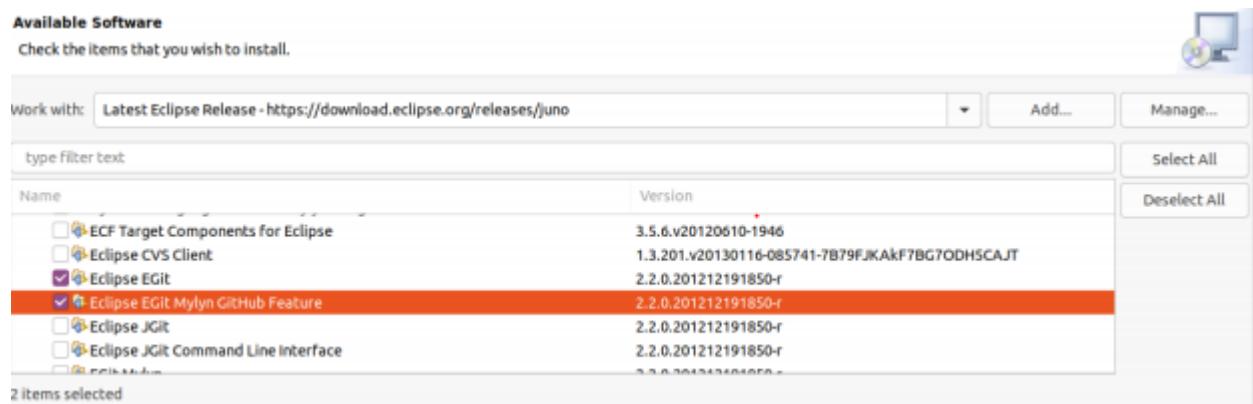


Рисунок 20 - Выбор модулей

Однако в новых версиях *Eclipse*, Git уже входит в состав среды.

Работа с git в Eclipse. Клонирование.

Выбираем пункты контекстного меню Window → Perspective → One Perspective → Other... (рис.21).

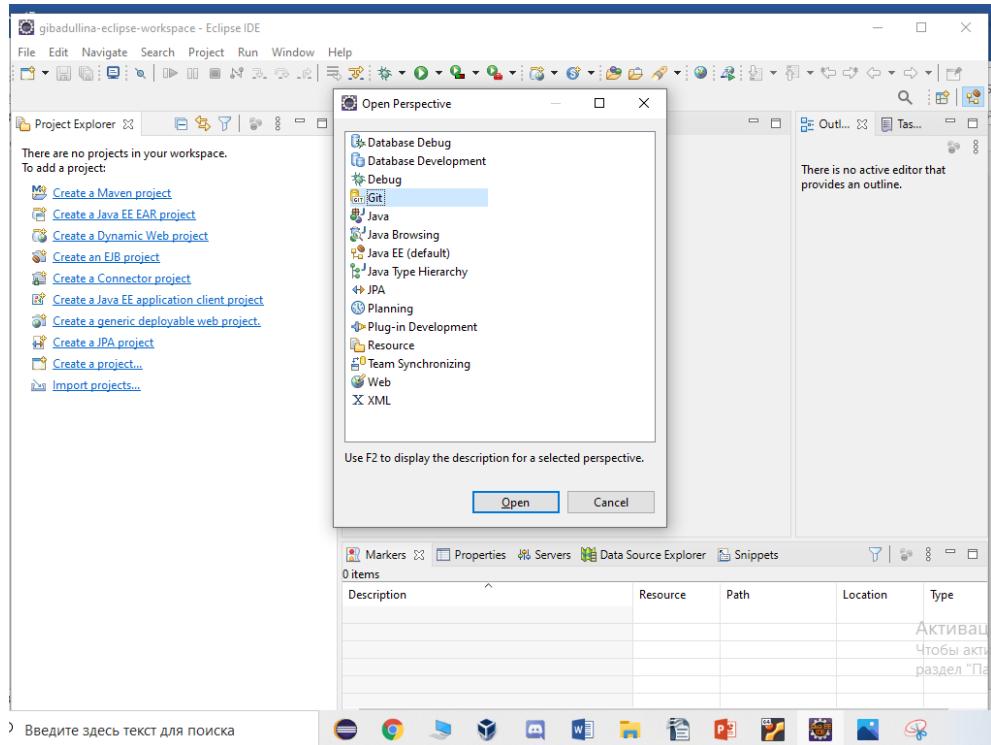


Рисунок 21 - Выбор перспективы

Выбирается проекция Git и на главном экране отобразиться пиктограмма Git. Клонирование репозитория и работа с git рассмотрены в следующем разделе.

### 3.2. Linux Ubuntu 20.04

Описание настройки операционной системы.

Для работы с инструментальной средой *Eclipse* в *Ubuntu* необходимо предварительно обновить систему:

- 1) Запустить терминал;
- 2) Выполнить команду `sudo apt update` для проверки доступных обновлений;
- 3) Для запуска процесса обновления выполнить команду `sudo apt upgrade`.

Результаты выполнения команд представлены на рисунке 22.

```
Ubuntu 20.04 [Работает] - Oracle VM VirtualBox
Файл Машина Вид Ввод Устройства Справка
Обзор Терминал Пн, 22 февраля 13:29 ru
gibadullina@gibadullina-VirtualBox: ~
gibadullina@gibadullina-VirtualBox:~$ sudo apt update
[sudo] пароль для gibadullina:
Сущ:1 http://ru.archive.ubuntu.com/ubuntu focal InRelease
Сущ:2 http://ru.archive.ubuntu.com/ubuntu focal-updates InRelease
Сущ:3 http://ru.archive.ubuntu.com/ubuntu focal-backports InRelease
Сущ:4 http://security.ubuntu.com/ubuntu focal-security InRelease
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
Может быть обновлено 15 пакетов. Запустите «apt list --upgradable» для их показа.
gibadullina@gibadullina-VirtualBox:~$ sudo apt upgrade
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
Расчёт обновлений... Готово
Следующие пакеты будут обновлены:
  dirmngr friendly-recovery gnupg gnupg-l10n gnupg-utils gpg gpg-agent
  gpg-wks-client gpg-wks-server gpgconf gpgsm gpgv
  python3-software-properties software-properties-common
  software-properties-gtk
Обновлено 15 пакетов, установлено 0 новых пакетов, для удаления отмечено 0 пакетов, и 0 пакетов не обновлено.
Необходимо скачать 0 B/2 672 kB архивов.
После данной операции объём занятого дискового пространства возрастёт на 0 B.
Хотите продолжить? [Д/Н] Д
(Чтение базы данных ... на данный момент установлено 149554 файла и каталога.)
Подготовка к распаковке .../00-gpg-wks-client_2.2.19-3ubuntu2.1_amd64.deb ...
Распаковывается gpg-wks-client (2.2.19-3ubuntu2.1) на замену (2.2.19-3ubuntu2)
```

Рисунок 22 - Обновление системы

Далее необходимо установить *JDK (OpenJDK8)*. Для этого выполняется команда в терминале: *sudo apt install openjdk-8-jdk* (рис.23)

Ubuntu 20.04 [Работает] - Oracle VM VirtualBox

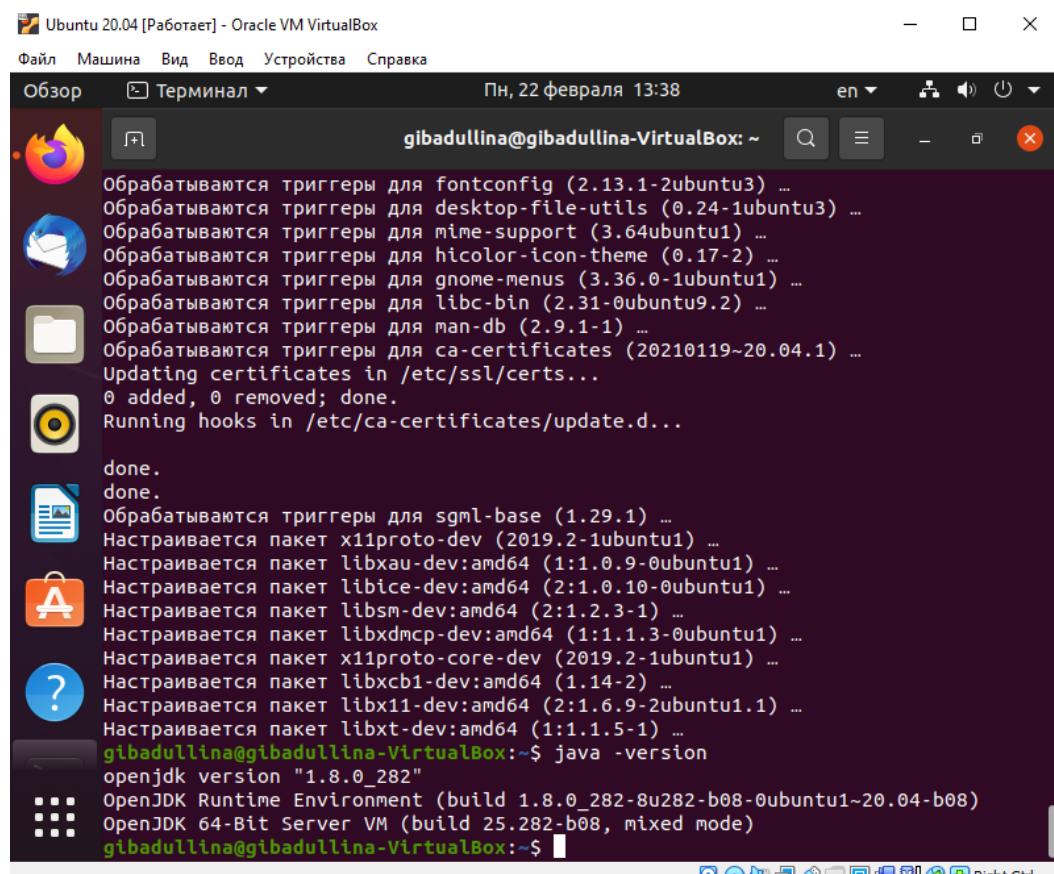
Файл Машина Вид Ввод Устройства Справка

Обзор Терминал Пн, 22 февраля 13:37 ru

```
gibadullina@gibadullina-VirtualBox: ~$ sudo apt install openjdk-8-jdk
Обрабатываются триггеры для man-db (2.9.1-1) ...
gibadullina@gibadullina-VirtualBox: ~$ sudo apt install openjdk-8-jdk
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
Будут установлены следующие дополнительные пакеты:
  ca-certificates-java fonts-dejavu-extra java-common libatk-wrapper-java
  libatk-wrapper-java-jni libice-dev libpthread-stubs0-dev libsm-dev
  libx11-dev libxau-dev libxcb1-dev libxdmcp-dev libxt-dev
  openjdk-8-jdk-headless openjdk-8-jre openjdk-8-jre-headless
  x11proto-core-dev x11proto-dev xorg-sgml-doctools xtrans-dev
Предлагаемые пакеты:
  default-jre libice-doc libsm-doc libx11-doc libxcb-doc libxt-doc
  openjdk-8-demo openjdk-8-source visualvm icedtea-8-plugin
  fonts-ipafont-gothic fonts-ipafont-mincho fonts-wqy-microhei
  fonts-wqy-zenhei
Следующие НОВЫЕ пакеты будут установлены:
  ca-certificates-java fonts-dejavu-extra java-common libatk-wrapper-java
  libatk-wrapper-java-jni libice-dev libpthread-stubs0-dev libsm-dev
  libx11-dev libxau-dev libxcb1-dev libxdmcp-dev libxt-dev openjdk-8-jdk
  openjdk-8-jdk-headless openjdk-8-jre openjdk-8-jre-headless
  x11proto-core-dev x11proto-dev xorg-sgml-doctools xtrans-dev
Обновлено 0 пакетов, установлено 21 новых пакетов, для удаления отмечено 0 пакетов, и 0 пакетов не обновлено.
Необходимо скачать 43,4 МВ архивов.
После данной операции объём занятого дискового пространства возрастёт на 162 МВ
.
Хотите продолжить? [Д/Н] Д
Пол:1 http://ru.archive.ubuntu.com/ubuntu focal/main amd64 java-common all 0.72
```

Рисунок 23 - Установка *OpenJDK 8*

После завершения установки необходимо проверить правильность установки *JDK* набрав команду *java -version*. В итоге в терминале должна быть выведена информация об установленной версии *JDK*. Результаты (рис. 24).



Ubuntu 20.04 [Работает] - Oracle VM VirtualBox

Файл Машина Вид Ввод Устройства Справка

Обзор Терминал Пн, 22 февраля 13:38 en

gibadullina@gibadullina-VirtualBox: ~

```
Обрабатываются триггеры для fontconfig (2.13.1-2ubuntu3) ...
Обрабатываются триггеры для desktop-file-utils (0.24-1ubuntu3) ...
Обрабатываются триггеры для time-support (3.64ubuntu1) ...
Обрабатываются триггеры для hicolor-icon-theme (0.17-2) ...
Обрабатываются триггеры для gnome-menus (3.36.0-1ubuntu1) ...
Обрабатываются триггеры для libc-bin (2.31-0ubuntu9.2) ...
Обрабатываются триггеры для man-db (2.9.1-1) ...
Обрабатываются триггеры для ca-certificates (20210119~20.04.1) ...
Updating certificates in /etc/ssl/certs...
0 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...

done.
done.
Обрабатываются триггеры для sgml-base (1.29.1) ...
Настраивается пакет x11proto-dev (2019.2-1ubuntu1) ...
Настраивается пакет libxau-dev:amd64 (1:1.0.9-0ubuntu1) ...
Настраивается пакет libice-dev:amd64 (2:1.0.10-0ubuntu1) ...
Настраивается пакет libsm-dev:amd64 (2:1.2.3-1) ...
Настраивается пакет libxdmcp-dev:amd64 (1:1.1.3-0ubuntu1) ...
Настраивается пакет x11proto-core-dev (2019.2-1ubuntu1) ...
Настраивается пакет libxcb1-dev:amd64 (1.14-2) ...
Настраивается пакет libx11-dev:amd64 (2:1.6.9-2ubuntu1.1) ...
Настраивается пакет libxt-dev:amd64 (1:1.1.5-1) ...
gibadullina@gibadullina-VirtualBox:~$ java -version
openjdk version "1.8.0_282"
OpenJDK Runtime Environment (build 1.8.0_282-8u282-b08-0ubuntu1~20.04-b08)
OpenJDK 64-Bit Server VM (build 25.282-b08, mixed mode)
gibadullina@gibadullina-VirtualBox:~$
```

Рисунок 24- Информация *JDK*

Установка *Eclipse IDE for Enterprise Java Developers*.

- 1) Использование браузера для загрузки дистрибутива Eclipse с официального сайта <https://www.eclipse.org/>.
- 2) Далее распаковка дистрибутива в домашнюю папку пользователя (рис.25)

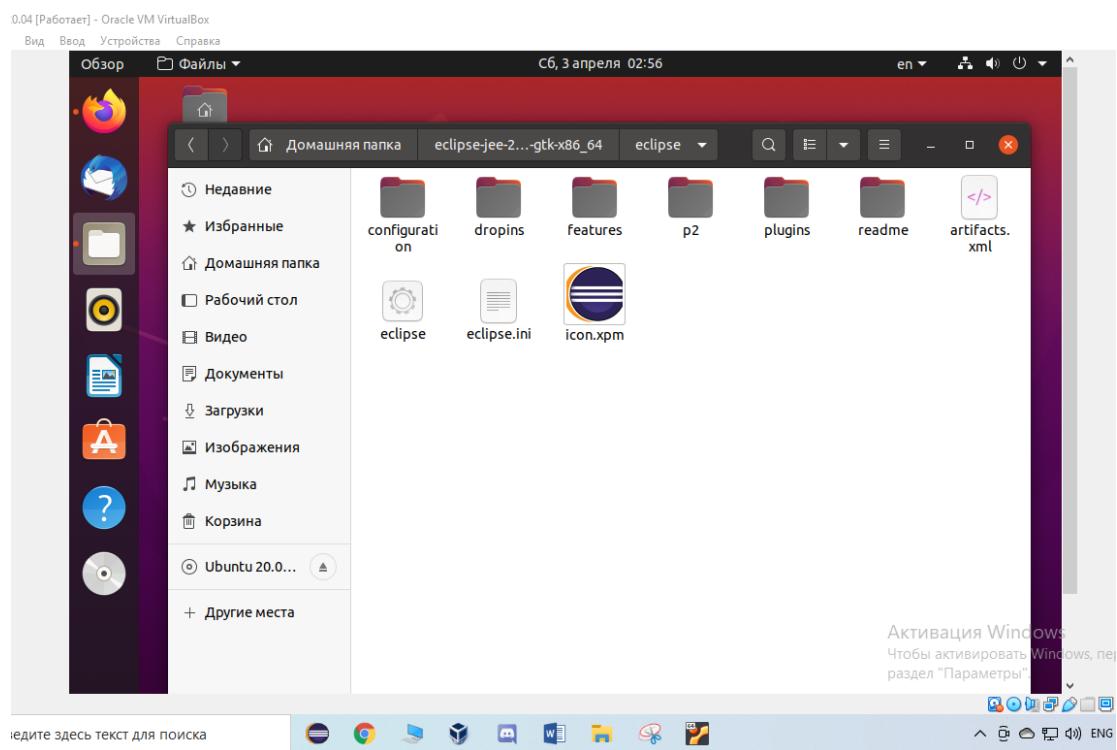


Рисунок 25- Распаковка

- 3) Выполнение файла eclipse из каталога eclipse.
- 4) Указывается рабочая область (место) на рисунке 26.

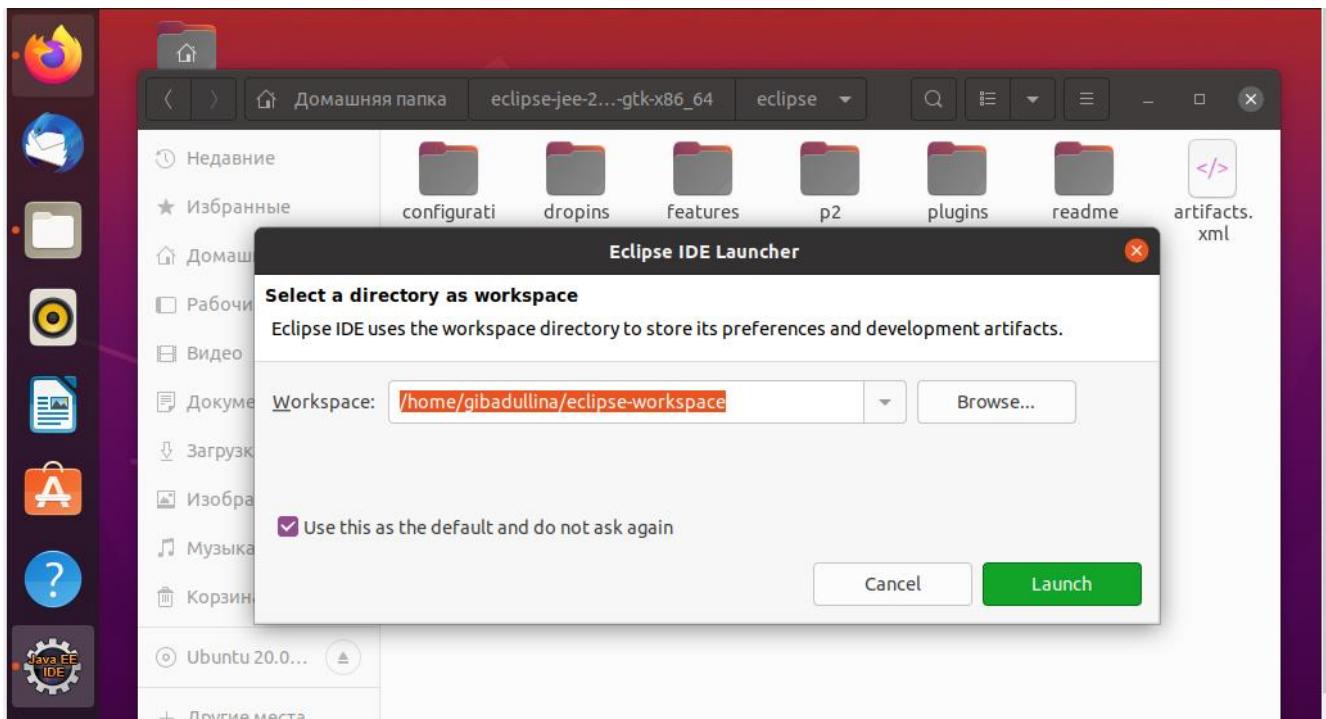


Рисунок 26- Workspace

- 5) Рабочее поле *Eclipse IDE for Enterprise Java Developers* (рис.27).

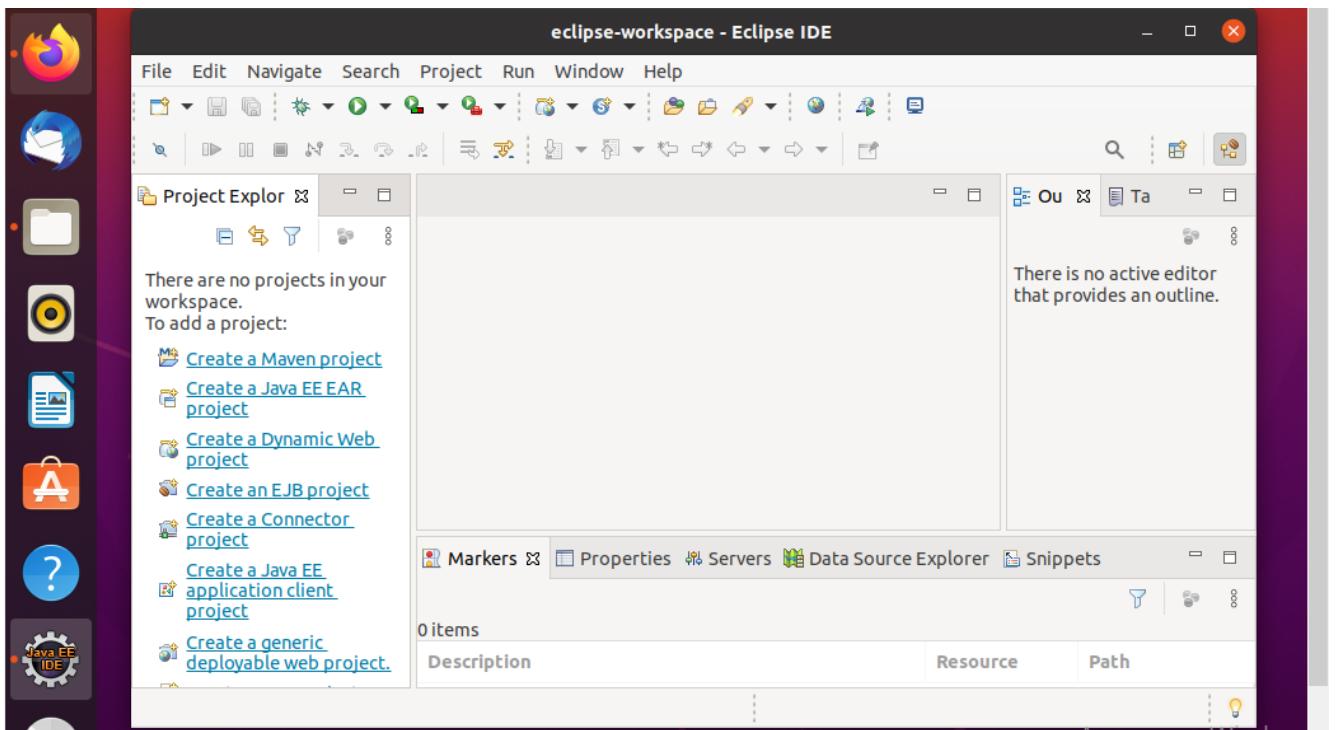


Рисунок 27- Eclipse IDE for Enterprise Java Developers

Установка Maven. В новых версиях Eclipse, Maven уже встроен в состав среды разработки.

Для создания Maven project выбираются пункты контекстного меню *File -> New -> Maven Project* (рис.28).

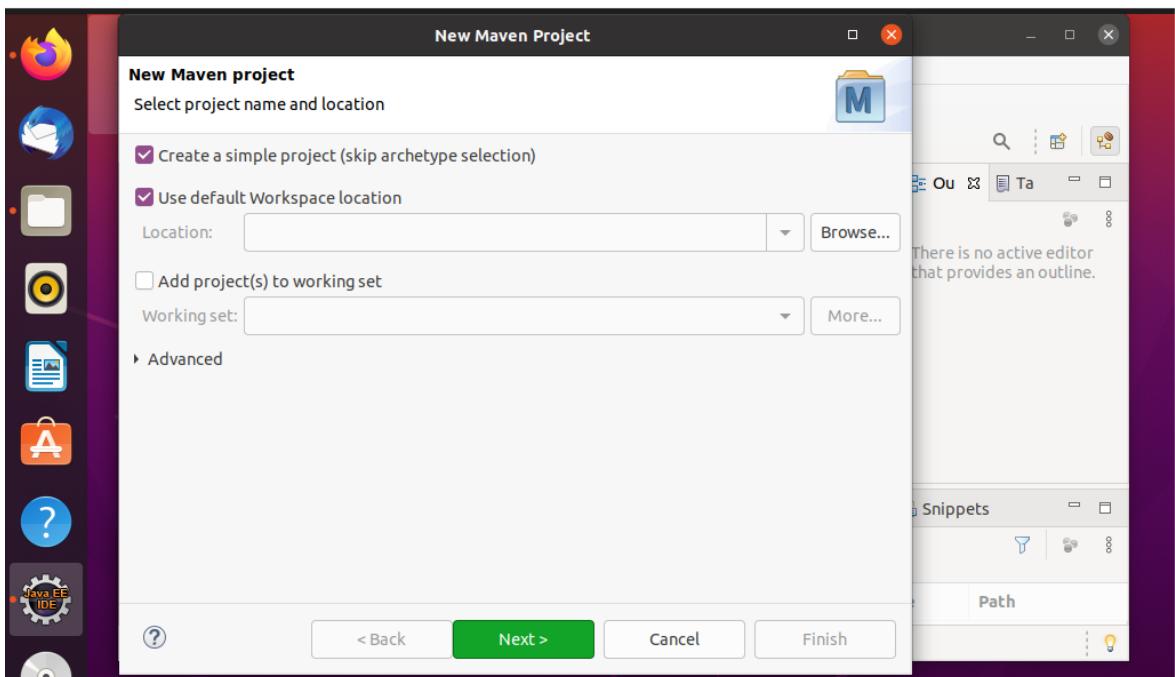


Рисунок 28- Создание нового проекта

Нажатие Next, переход к настройке проекта (рис.29)

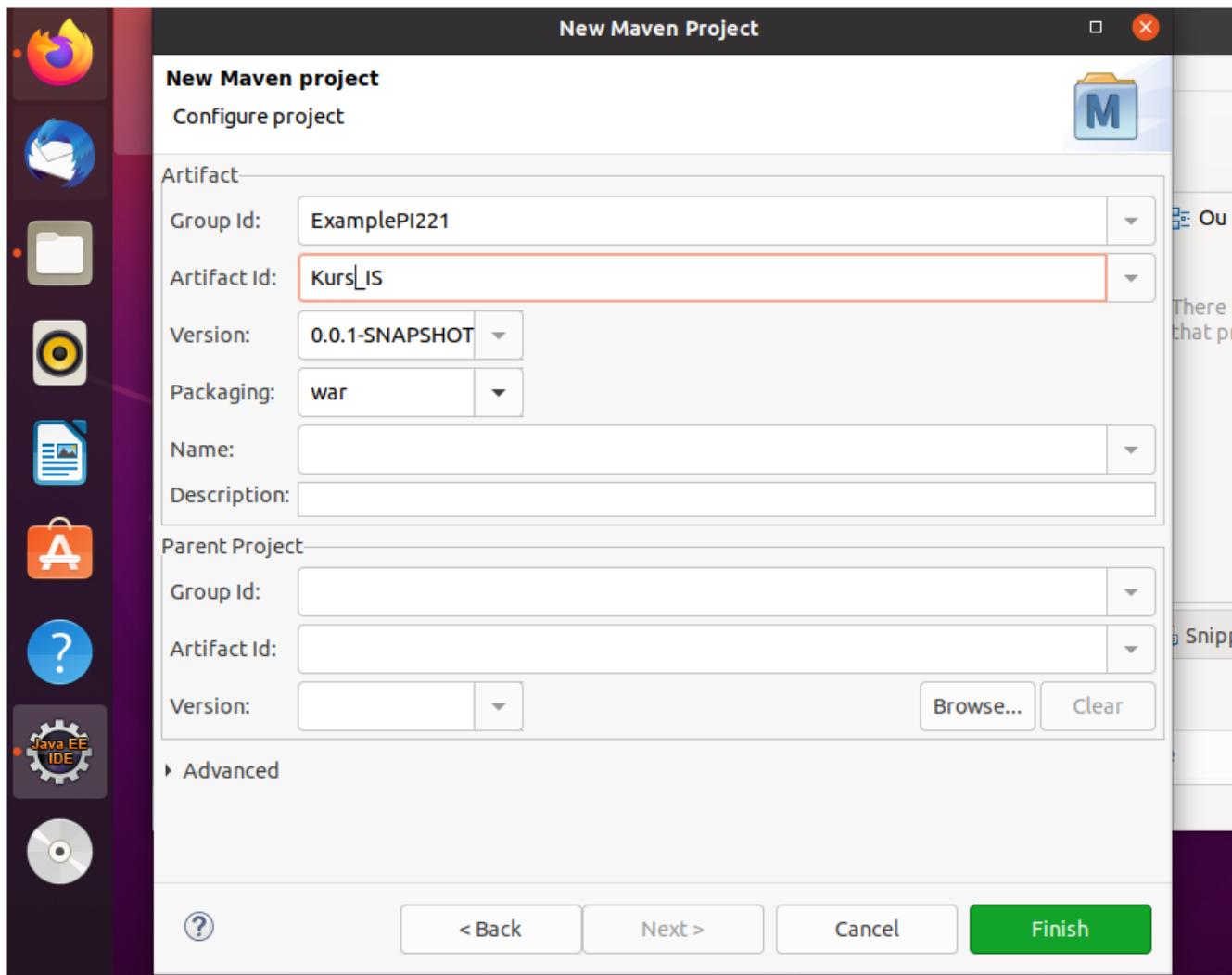


Рисунок 29- Настройка проекта

В итоге создан Maven Project (рис.30).

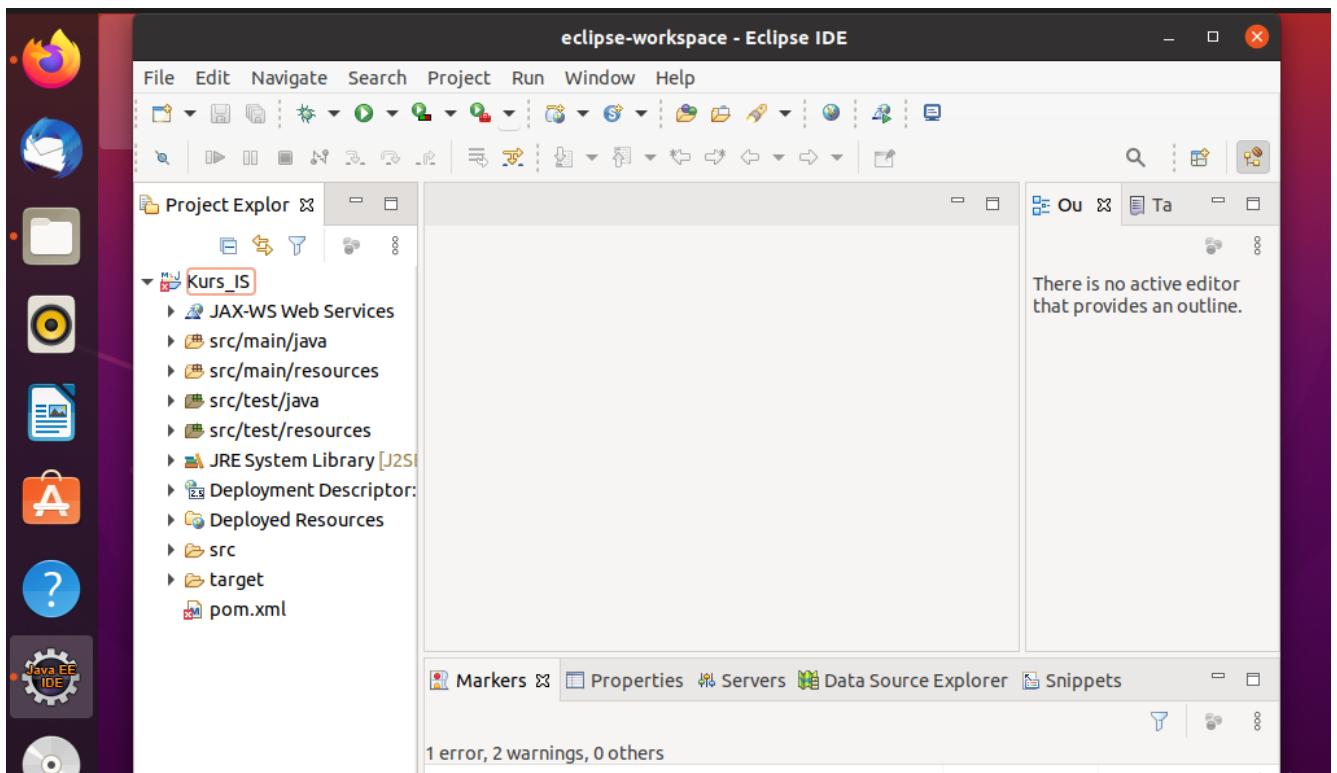


Рисунок 30 - Проект Maven

Установка Git.

В новых версиях Eclipse, Git уже входит в состав среды.

Работа с git в Eclipse. Клонирование.

Выбираем пункты контекстного меню Window → Perspective → One Perspective → Other... (рис.31).

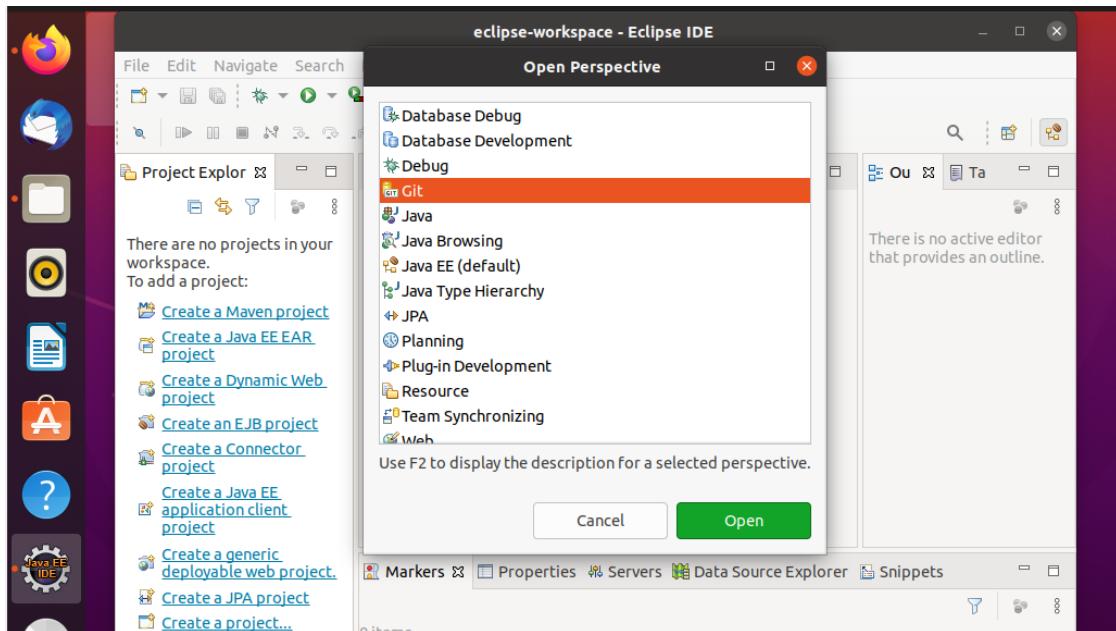


Рисунок 31- Выбор перспективы

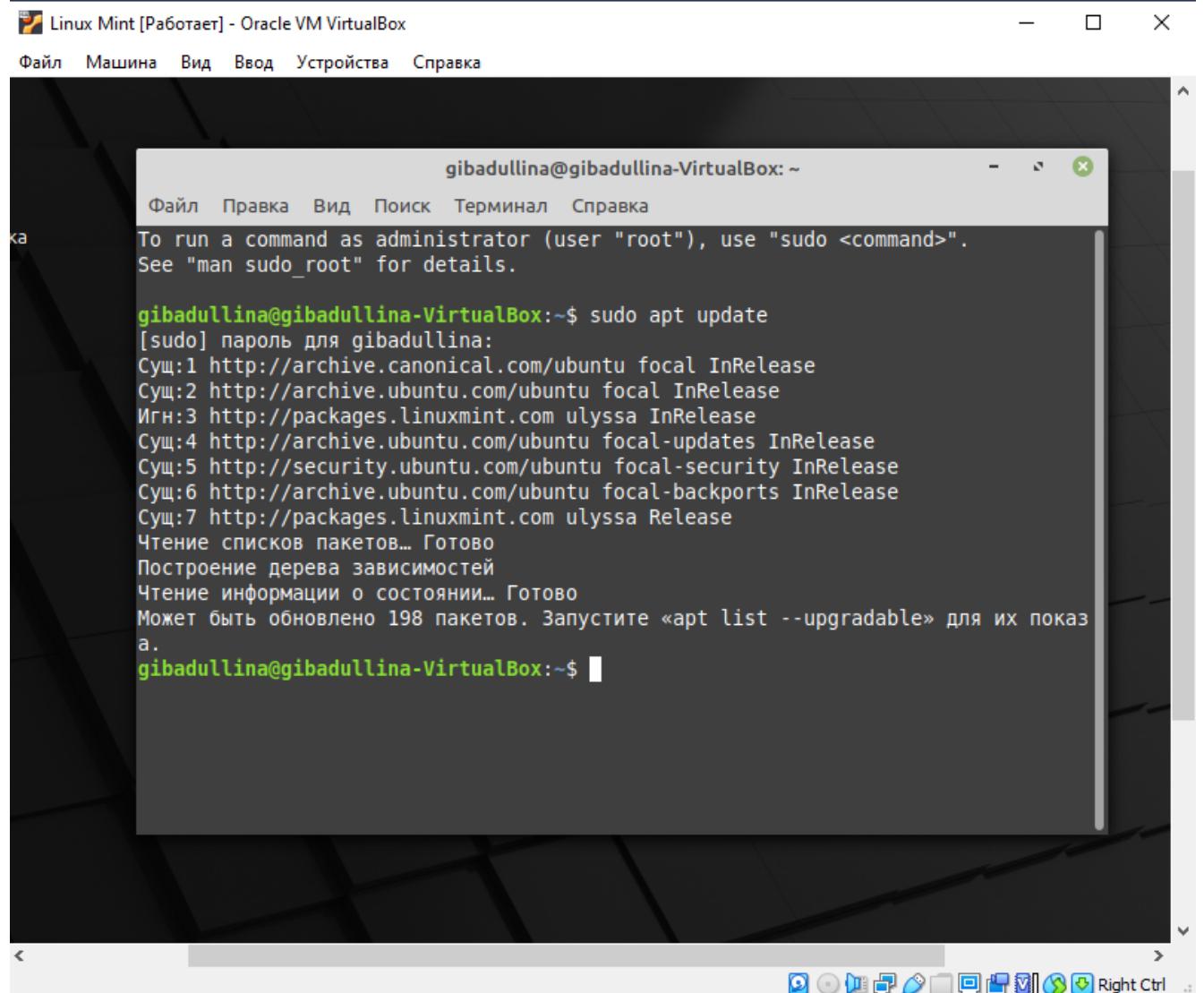
Выбирается проекция Git и на главном экране отобразиться пиктограмма Git. Клонирование репозитория и работа с git рассмотрены в следующем разделе.

### 3.3. Linux Mint

Описание настройки операционной системы.

Для работы с инструментальной средой *Eclipse* в *Mint* необходимо предварительно обновить систему:

- 1) Запустить терминал;
- 2) Выполнить команду *sudo apt update* для проверки доступных обновлений (рис.32);



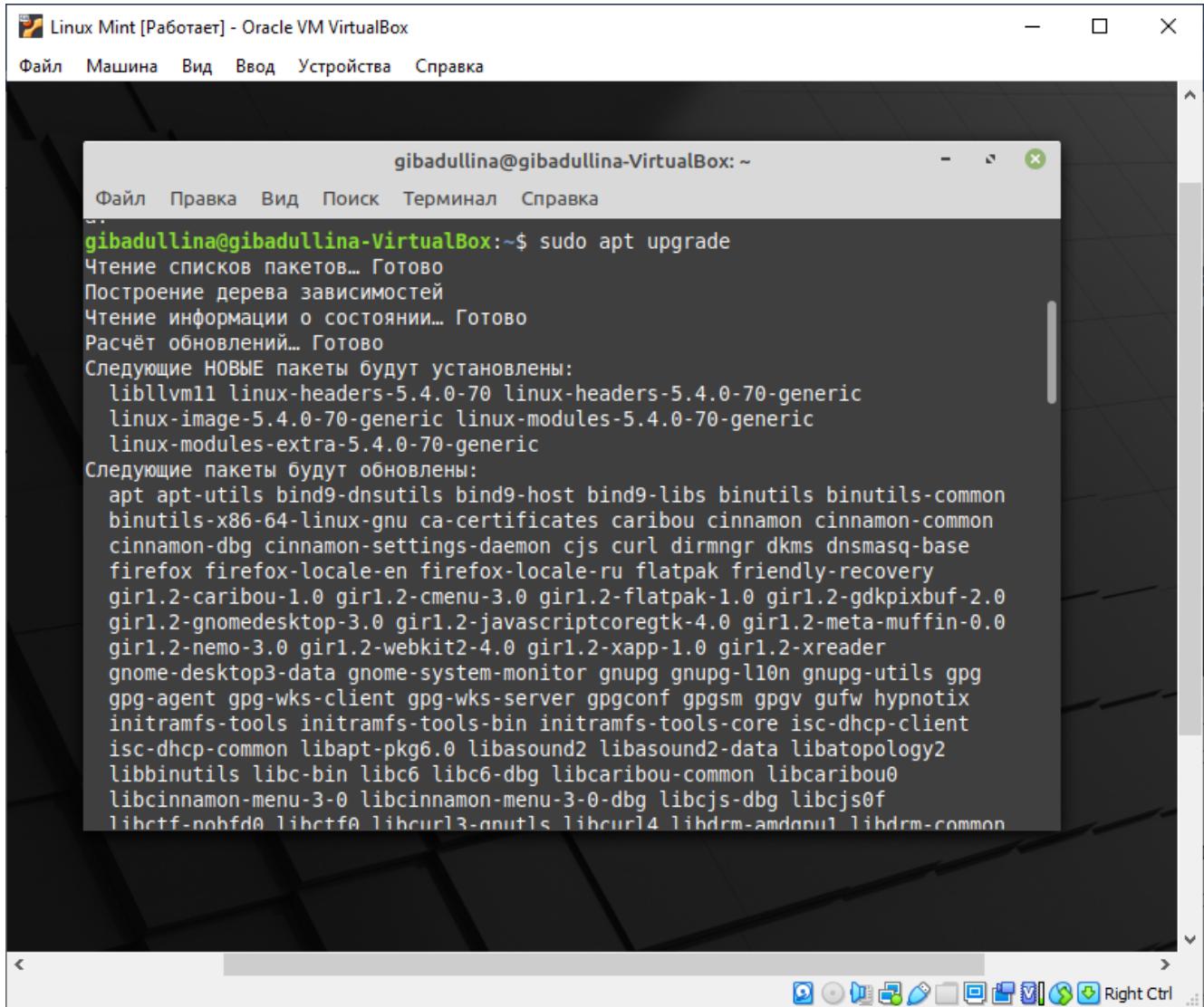
```
gibadullina@gibadullina-VirtualBox: ~
Файл Правка Вид Поиск Терминал Справка
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

gibadullina@gibadullina-VirtualBox:~$ sudo apt update
[sudo] пароль для gibadullina:
Сущ:1 http://archive.canonical.com/ubuntu focal InRelease
Сущ:2 http://archive.ubuntu.com/ubuntu focal InRelease
Игн:3 http://packages.linuxmint.com ulyssa InRelease
Сущ:4 http://archive.ubuntu.com/ubuntu focal-updates InRelease
Сущ:5 http://security.ubuntu.com/ubuntu focal-security InRelease
Сущ:6 http://archive.ubuntu.com/ubuntu focal-backports InRelease
Сущ:7 http://packages.linuxmint.com ulyssa Release
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
Может быть обновлено 198 пакетов. Запустите «apt list --upgradable» для их показа.
gibadullina@gibadullina-VirtualBox:~$
```

Рисунок 32— Проверка обновлений системы

3) Для запуска процесса обновления выполнить команду *sudo apt upgrade*.

Результаты выполнения команд представлены на рисунке 33.



The screenshot shows a terminal window titled "Linux Mint [Работает] - Oracle VM VirtualBox". The window has a menu bar with "Файл", "Машина", "Вид", "Ввод", "Устройства", and "Справка". Below the menu is a toolbar with icons for "Файл", "Правка", "Вид", "Поиск", "Терминал", and "Справка". The terminal itself has a title bar "gibadullina@gibadullina-VirtualBox: ~" and a menu bar with "Файл", "Правка", "Вид", "Поиск", "Терминал", and "Справка". The main area of the terminal displays the output of the command "sudo apt upgrade". The output shows the process of reading package lists, building dependency trees, and calculating upgrade packages. It lists packages to be installed (libl1v11, linux-headers-5.4.0-70, etc.) and packages to be updated (apt, bind9-dnsutils, bind9-host, etc.). The terminal window is set against a dark background of the desktop environment.

```
gibadullina@gibadullina-VirtualBox:~$ sudo apt upgrade
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
Расчёт обновлений... Готово
Следующие НОВЫЕ пакеты будут установлены:
  libl1v11 linux-headers-5.4.0-70 linux-headers-5.4.0-70-generic
  linux-image-5.4.0-70-generic linux-modules-5.4.0-70-generic
  linux-modules-extra-5.4.0-70-generic
Следующие пакеты будут обновлены:
  apt apt-utils bind9-dnsutils bind9-host bind9-libs binutils binutils-common
  binutils-x86-64-linux-gnu ca-certificates caribou cinnamon cinnamon-common
  cinnamon-dbg cinnamon-settings-daemon cjs curl dirmngr dkms dnsmasq-base
  firefox firefox-locale-en firefox-locale-ru flatpak friendly-recovery
  gir1.2-caribou-1.0 gir1.2-cmenu-3.0 gir1.2-flatpak-1.0 gir1.2-gdkpixbuf-2.0
  gir1.2-gnomedesktop-3.0 gir1.2-javascriptcoregtk-4.0 gir1.2-meta-muffin-0.0
  gir1.2-nemo-3.0 gir1.2-webkit2-4.0 gir1.2-xapp-1.0 gir1.2-xreader
  gnome-desktop3-data gnome-system-monitor gnupg gnupg-l10n gnupg-utils gpg
  gpg-agent gpg-wks-client gpg-wks-server gpgconf gpgsm gpgv gufw hypnotix
  initramfs-tools initramfs-tools-bin initramfs-tools-core isc-dhcp-client
  isc-dhcp-common libapt-pkg6.0 libasound2 libasound2-data libatopology2
  libbinutils libc-bin libc6 libc6-dbg libcaribou-common libcaribou0
  libcinnamon-menu-3-0 libcinnamon-menu-3-0-dbg libcjsg-dbg libcjsg0f
  libctf-nohfd0 libctf0 libcurl3-gnutls libcurl4 libdrm-amd0u1 libdrm-common
```

Рисунок 33- Обновление системы

Далее необходимо установить *JDK (OpenJDK8)*. Для этого выполняется команда в терминале: *sudo apt install openjdk-8-jdk*. После завершения установки необходимо проверить правильность установки *JDK* набрав команду *java -version*. В итоге в терминале должна быть выведена информация об установленной версии *JDK*. Результаты (рис.34).

Linux Mint [Работает] - Oracle VM VirtualBox

Файл Машина Вид Ввод Устройства Справка

```
gibadullina@gibadullina-VirtualBox:~$ sudo apt install openjdk-8-jdk
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
Будут установлены следующие дополнительные пакеты:
  fonts-dejavu-extra libatk-wrapper-java libatk-wrapper-java-jni libice-dev
  libpthread-stubs0-dev libsm-dev libx11-dev libxau-dev libxcb1-dev
  libxdmcp-dev libxt-dev openjdk-8-jdk-headless openjdk-8-jre
  openjdk-8-jre-headless x11proto-core-dev x11proto-dev xorg-sgml-doctools
  xtrans-dev
Предлагаемые пакеты:
  libice-doc libsm-doc libx11-doc libxcb-doc libxt-doc openjdk-8-demo
  openjdk-8-source visualvm icedtea-8-plugin fonts-ipafont-gothic
  fonts-ipafont-mincho fonts-wqy-microhei fonts-wqy-zenhei
Следующие НОВЫЕ пакеты будут установлены:
  fonts-dejavu-extra libatk-wrapper-java libatk-wrapper-java-jni libice-dev
  libpthread-stubs0-dev libsm-dev libx11-dev libxau-dev libxcb1-dev
  libxdmcp-dev libxt-dev openjdk-8-jdk openjdk-8-jdk-headless openjdk-8-jre
  openjdk-8-jre-headless x11proto-core-dev x11proto-dev xorg-sgml-doctools
  xtrans-dev
Обновлено 0 пакетов, установлено 19 новых пакетов, для удаления отмечено 0 пакетов, и 0 пакетов не обновлено.
Необходимо скачать 43,4 МВ архивов.
После данной операции объём занятого дискового пространства возрастёт на 162 МВ.
```

Рисунок 34- Установка *OpenJDK 8*

После завершения установки необходимо проверить правильность установки *JDK* набрав команду *java -version*. В итоге в терминале должна быть выведена информация об установленной версии *JDK*. Результаты (рис.35).

gibadullina@gibadullina-VirtualBox:~\$

Файл Правка Вид Поиск Терминал Справка

```
gibadullina@gibadullina-VirtualBox:~$ java -version
openjdk version "11.0.10" 2021-01-19
OpenJDK Runtime Environment (build 11.0.10+9-Ubuntu-0ubuntu1.20.04)
OpenJDK 64-Bit Server VM (build 11.0.10+9-Ubuntu-0ubuntu1.20.04, mixed mode)
```

Рисунок 35- Информация *JDK*

Установка *Eclipse*.

- 6) Использование браузера для загрузки дистрибутива Eclipse с официального сайта <https://www.eclipse.org/>.
- 7) Далее распаковка дистрибутива в домашнюю папку пользователя (рис.36)

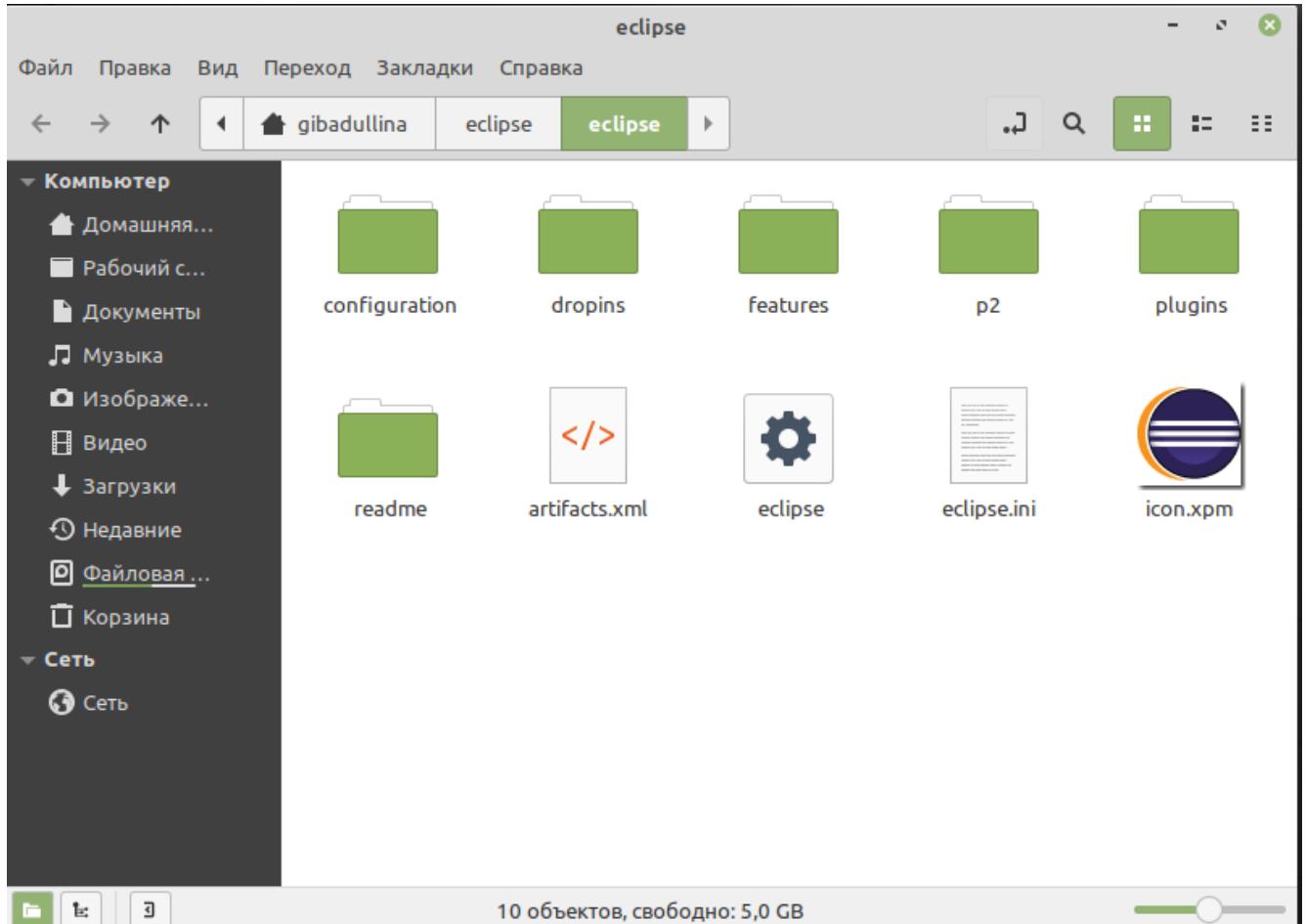


Рисунок 36- Распаковка

- 8) Выполнение файла eclipse из каталога eclipse.
- 9) Указывается рабочая область (место) на рисунке 37.

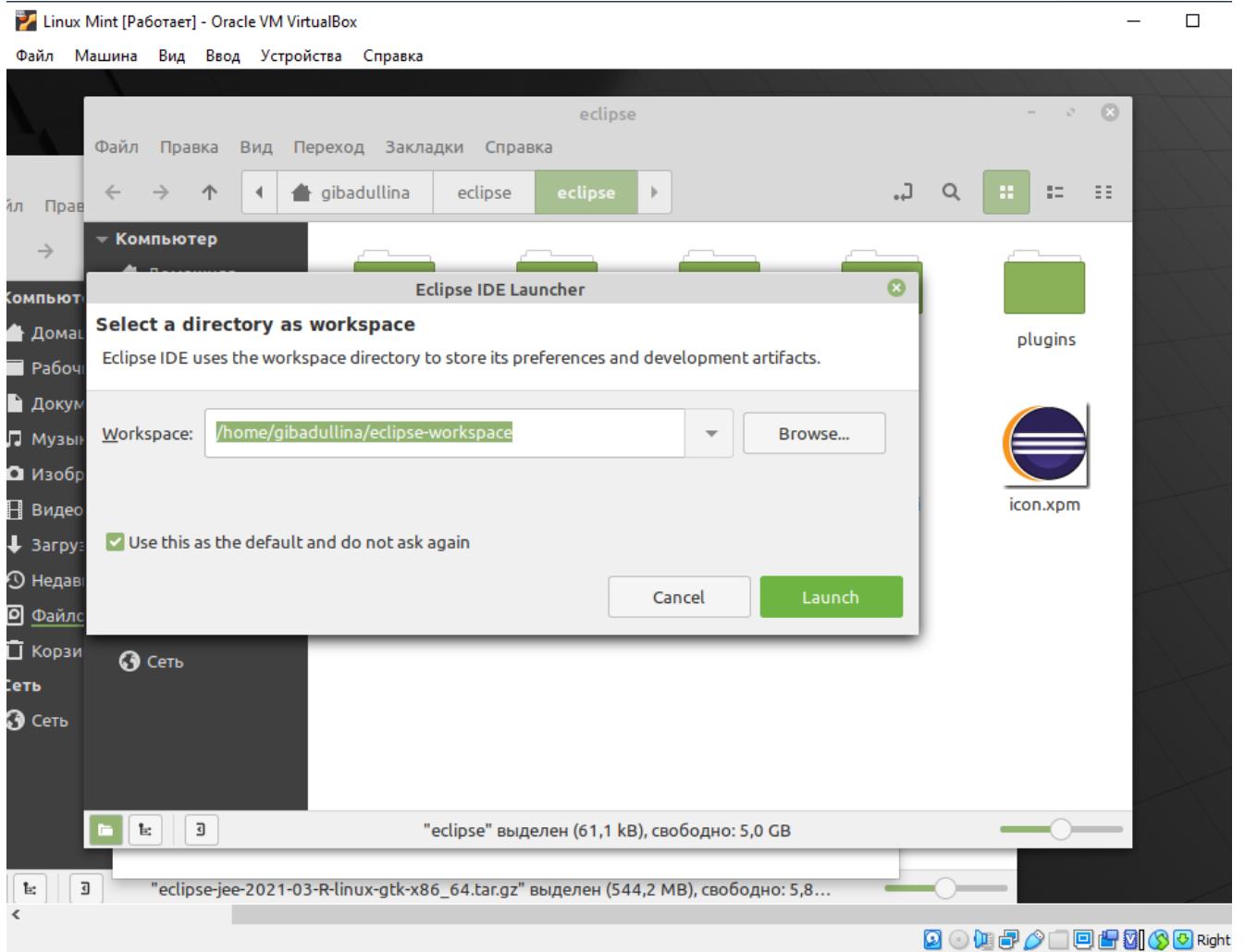


Рисунок 37- Указание workspace

10) Запуск Eclipse (рис.38).

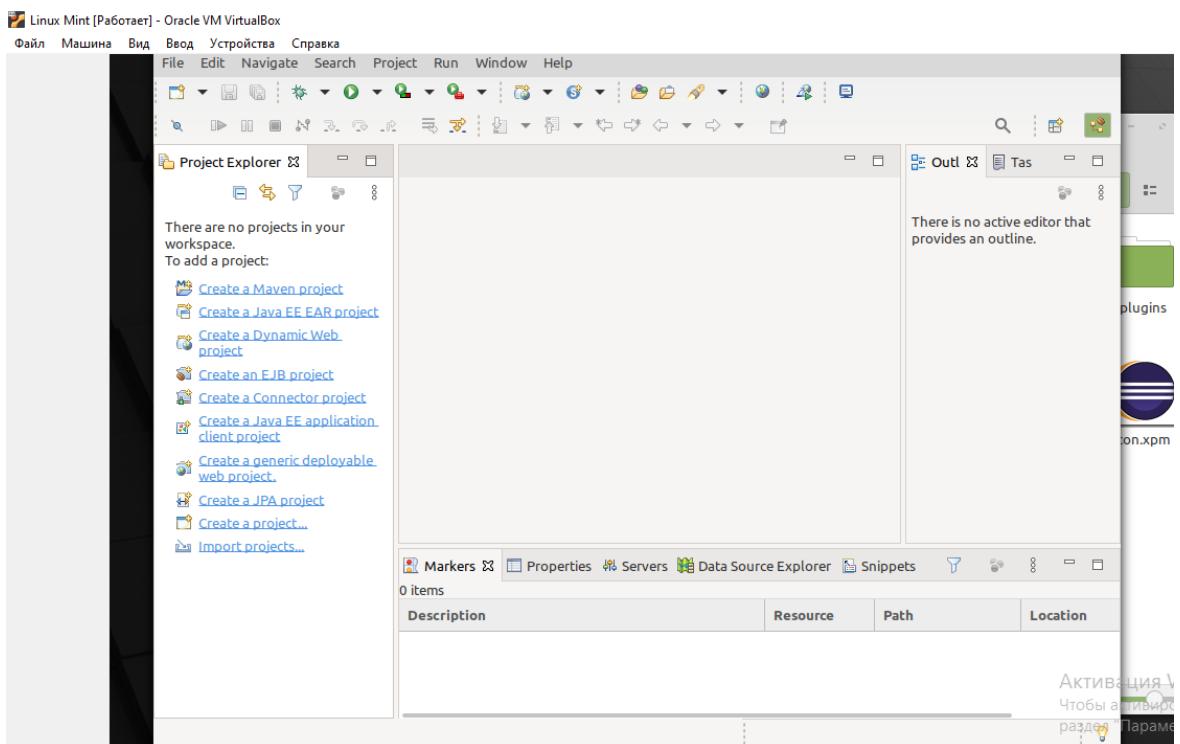


Рисунок 38- Рабочая область Eclipse

Установка Maven. В новых версиях Eclipse, Maven уже встроен в состав среды разработки.

Для создания Maven project выбираются пункты контекстного меню *File -> New -> Maven Project* (рис.39).

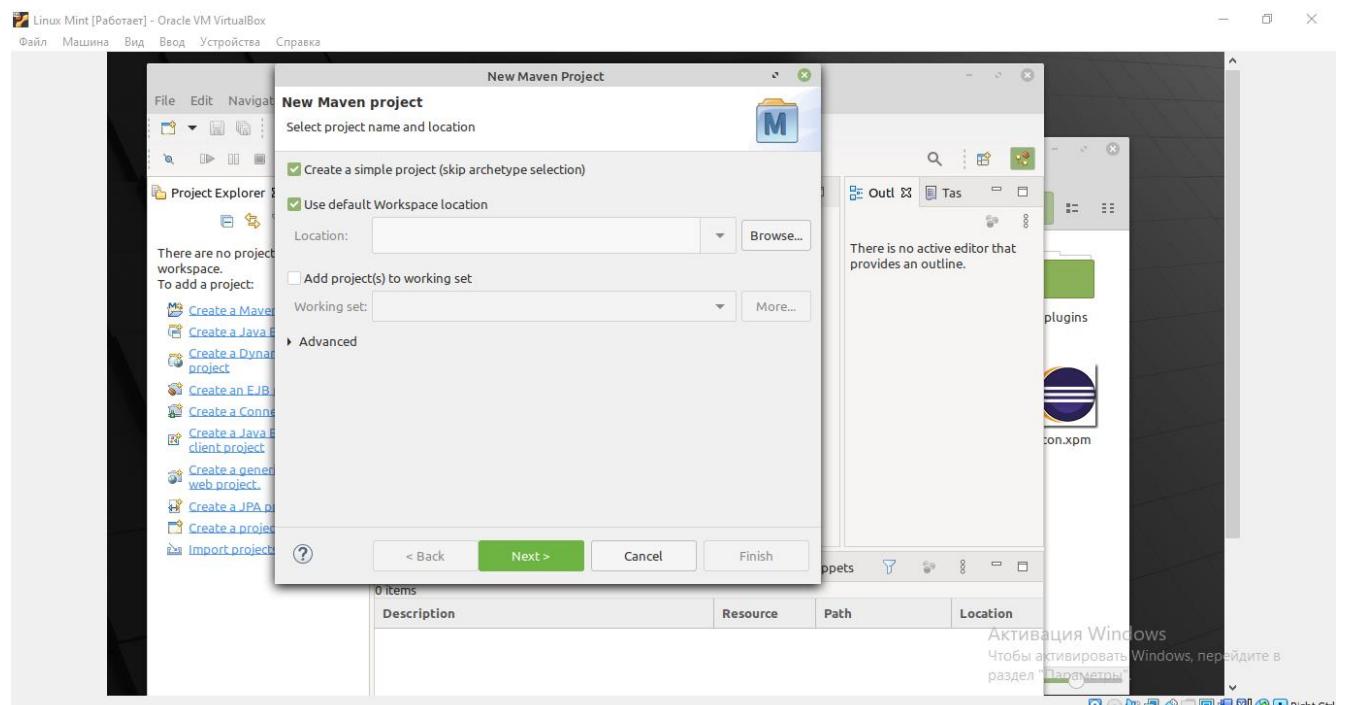


Рисунок 39- Создание нового проекта

Нажатие Next, переход к настройке проекта (рис.40)

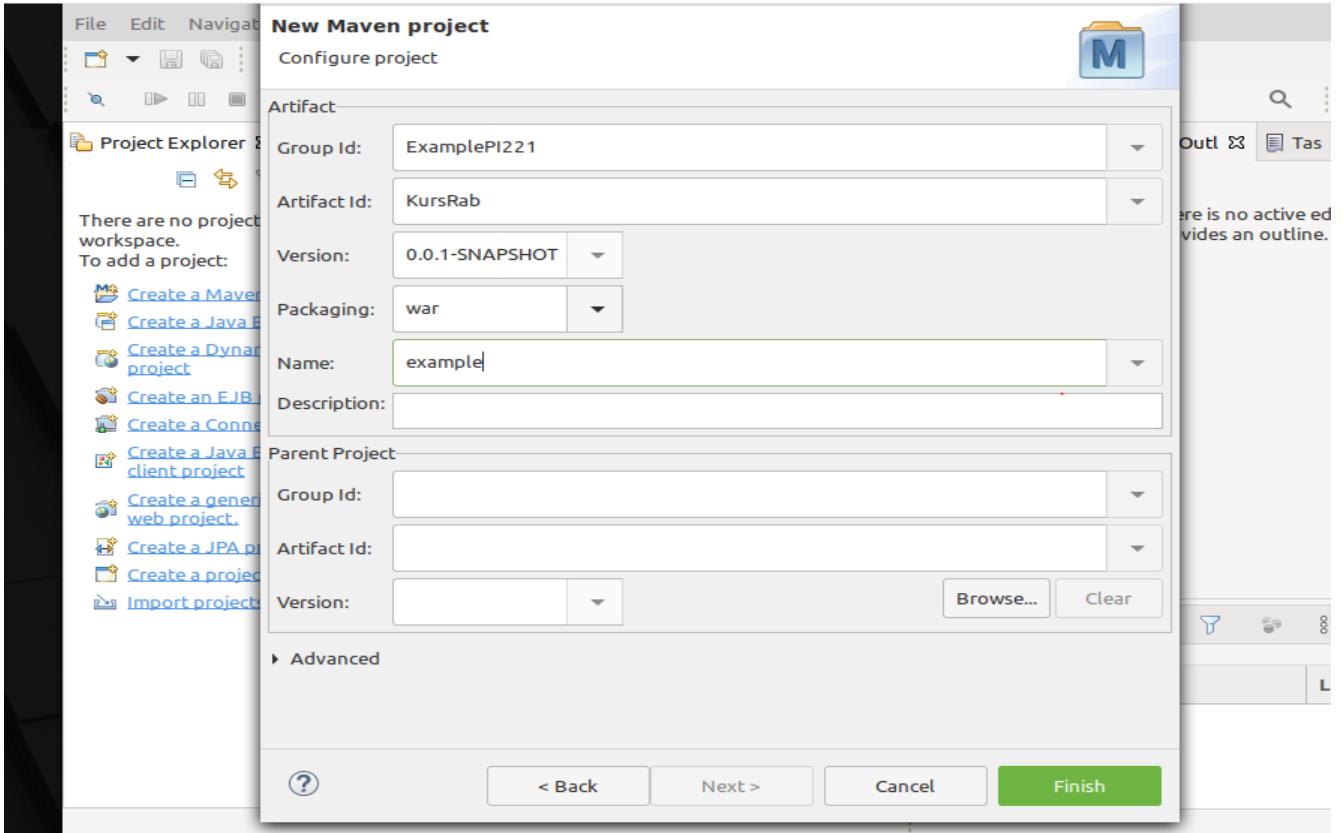


Рисунок 40- Настройка проекта

В итоге создан Maven Project (рис.41).

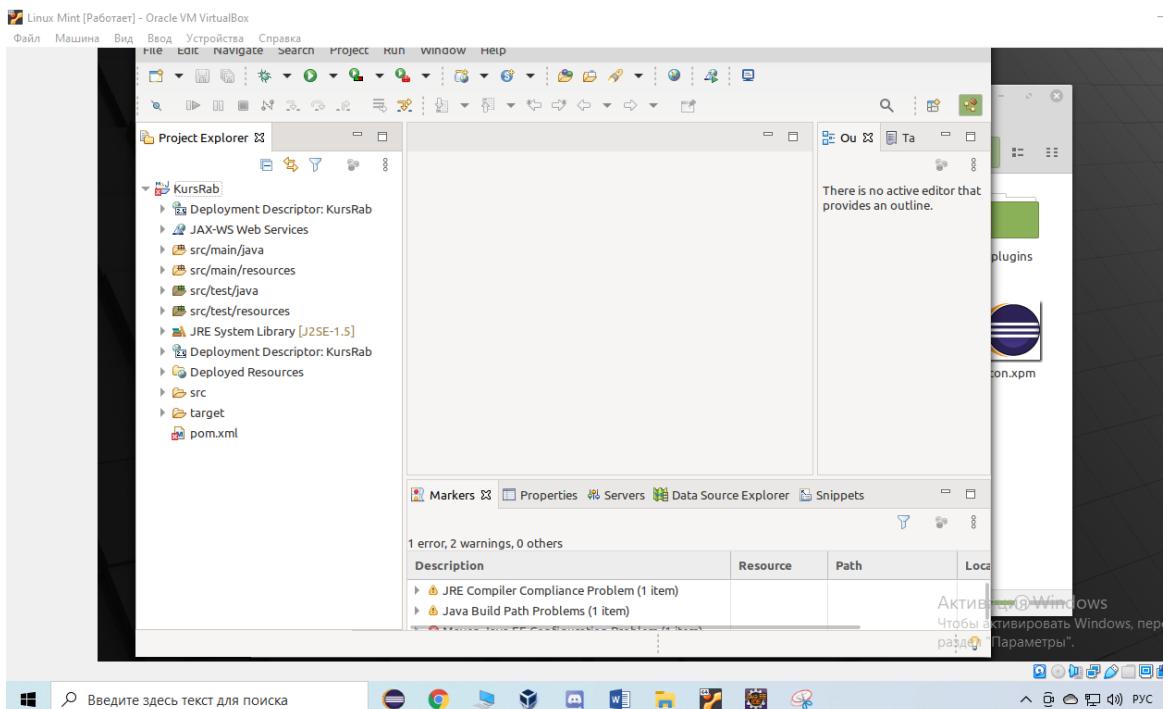


Рисунок 41- Проект Maven

Установка Git.

В новых версиях Eclipse, Git уже входит в состав среды.

## Работа с git в Eclipse.

Выбираем пункты контекстного меню Window → Perspective → One Perspective → Other... (рис.42).

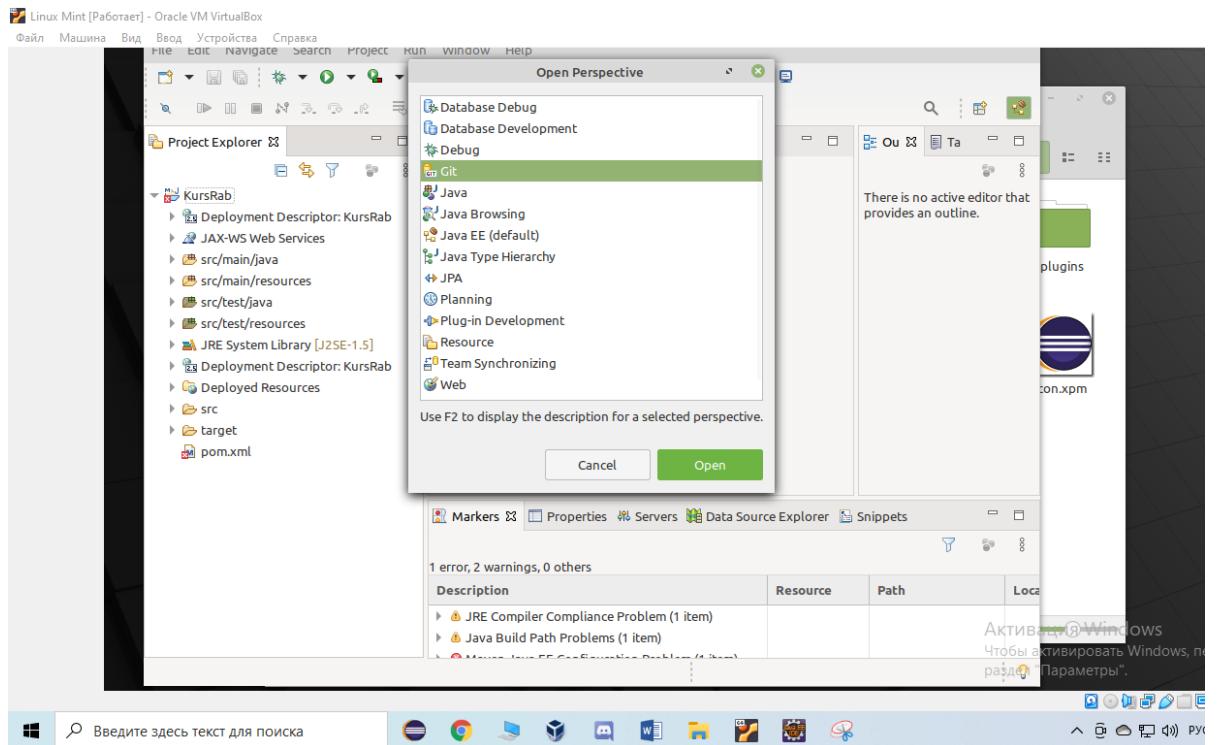


Рисунок 42- Выбор перспективы

Выбирается проекция Git и на главном экране отобразиться пиктограмма Git. Клонирование репозитория и работа с git рассмотрены в следующем разделе.

## **Раздел 4. Настройка среды разработки для подключения к системе контроля версий**

В данном разделе курсовой работы представлена инструкция для подключения разработчика к системе контроля версий *Git*, где в качестве хостинга выступает веб-сервис *GitHub*.

Среда разработки Eclipse универсальна, поэтому наглядная демонстрация этапов настройки и подключения разработчика будет показана в ОС *Linux Ubuntu 20.04*.

Инструкция:

- 1) Ссылка на репозиторий *GitHub*:

<https://github.com/Gibadullina/calculator-team1>

- 2) Настройка *Eclipse IDE* для работы с *GitHub*.

*Eclipse* для работы с [github.com](http://github.com) необходимы дополнительно подключаемые модули, в частности «*Git*». Необходимо в меню выбрать «*Help*», затем «*Install New Software...*» и далее в поле «*Work with:*» ввести адрес: <http://download.eclipse.org/releases/juno>.

Далее необходимо выбрать «*Collaboration*», и отметить модули «*Eclipse EGit*» и «*Eclipse EGit Mylyn Github Feature*» (рис. 43).

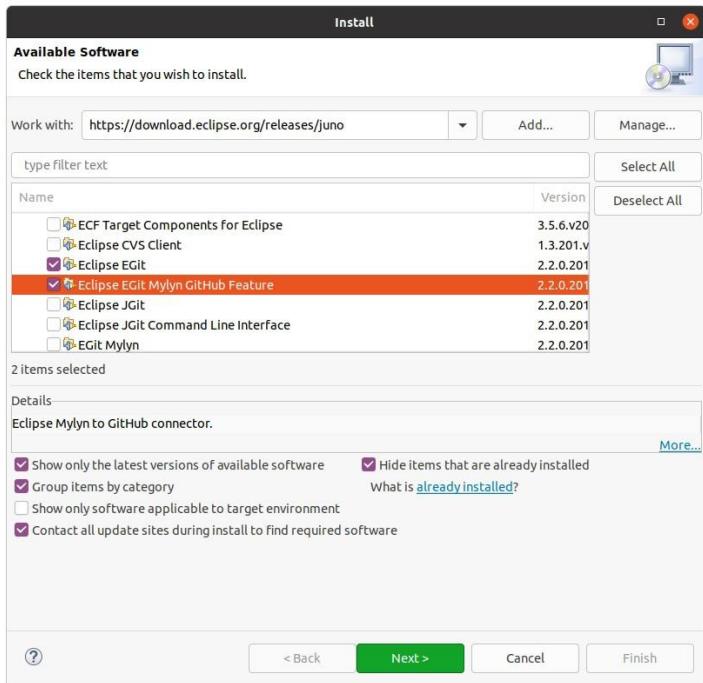


Рисунок 43 - Выбор модулей на установку в *Eclipse*

После того, как были выбраны элементы, нажимаем «*Next*» и переходим к просмотру элементов, которые необходимо установить (рис.44). Убедимся, что устанавливаются необходимые элементы и нажимаем «*Next*».

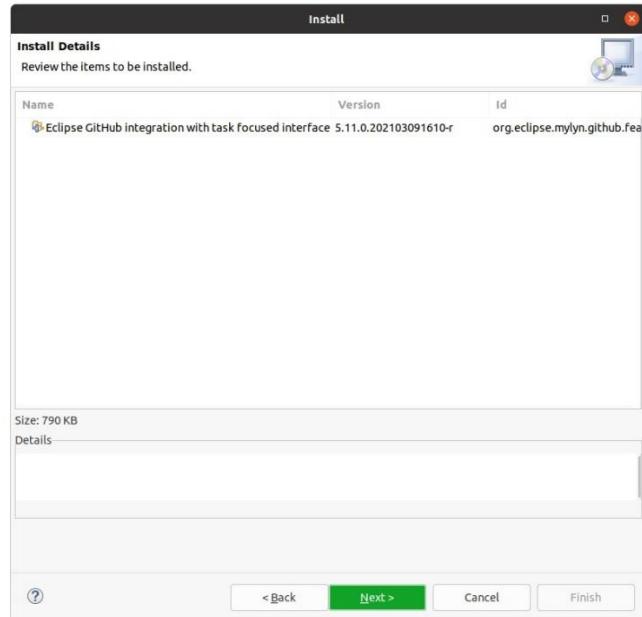


Рисунок 44 - Вкладка *Install Details*

Потом необходимо изучить и принять лицензионное соглашение (рис.45).

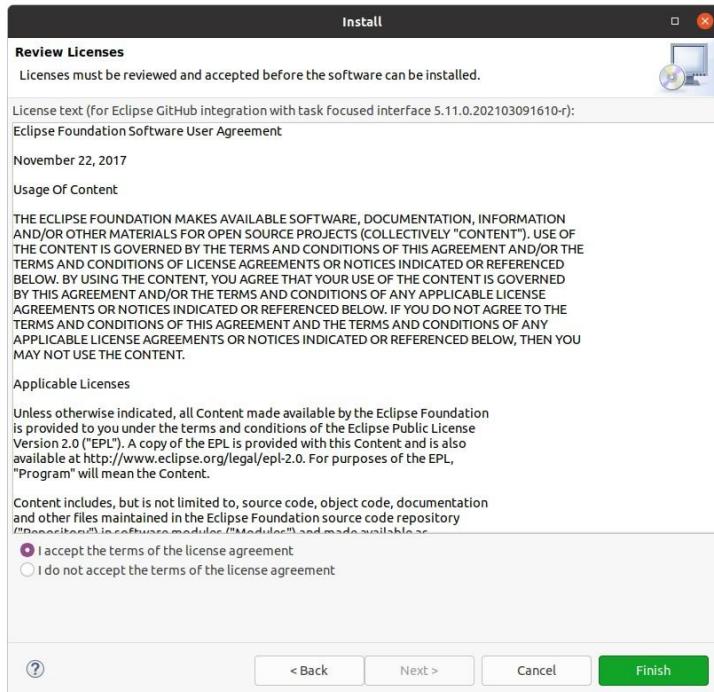


Рисунок 45 - Лицензионное соглашение

Далее следует нажать «*Finish*», после завершения установки происходит перезагрузка *Eclipse*.

Переход в проекцию работы с *git* из меню «*Window*» → «*Perspective*» → «*Open Perspective*» → «*Other...*» (рис.46). В открывшемся окне выбираем «*Git*» и нажимаем «*Open*».

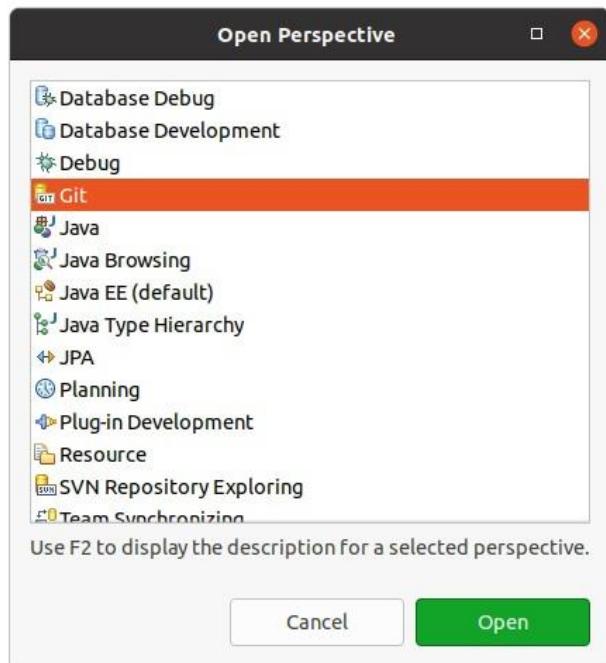


Рисунок 46 - Окно *Open Perspective*

Теперь на главной странице *Eclipse* присутствует проекция *Git* (рис.47).

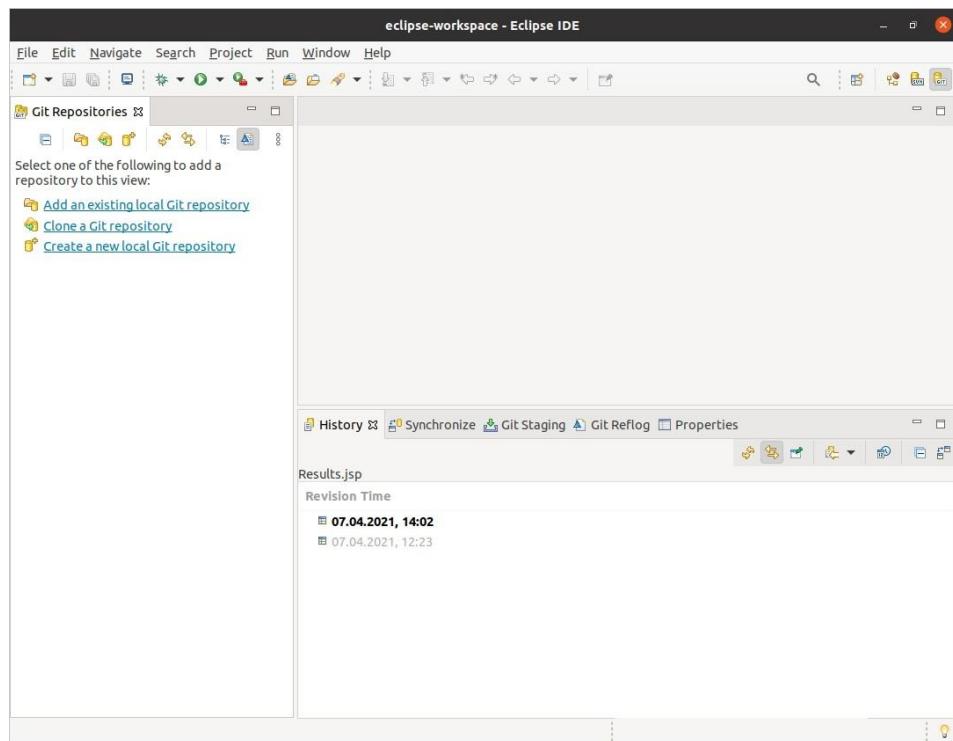


Рисунок 47 - Проекция *Git*

### 3) Выполнение клонирования репозитория.

Модуль *Eclipse* предлагает три действия: добавить существующий локальный репозиторий, клонировать репозиторий и создать новый локальный репозиторий. Чтобы клонировать репозиторий, необходимо нажать на «*Clone a Git repository*».

Далее откроется окно выбора расположения репозитория *Git*. Указываем в открывшемся окне *URI, Host (github.com)* и путь к репозиторию, а также если нужно, то имя пользователя и пароль (рис.48). Нажать «*Next*».

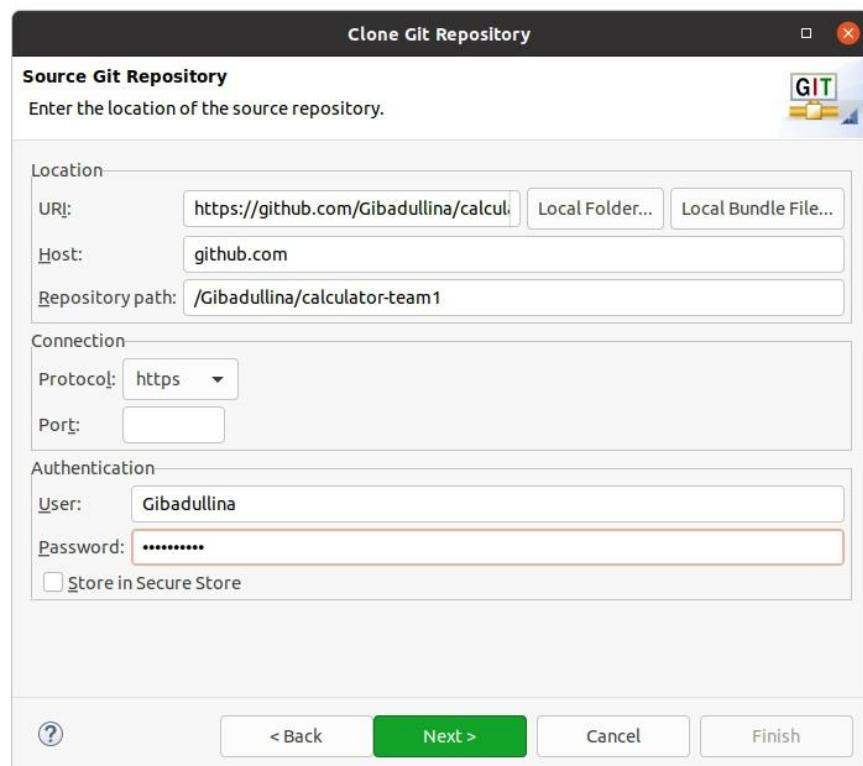


Рисунок 48 - Клонирование репозитория

На следующем шаге необходимо выбрать нужную ветку (рис.49).

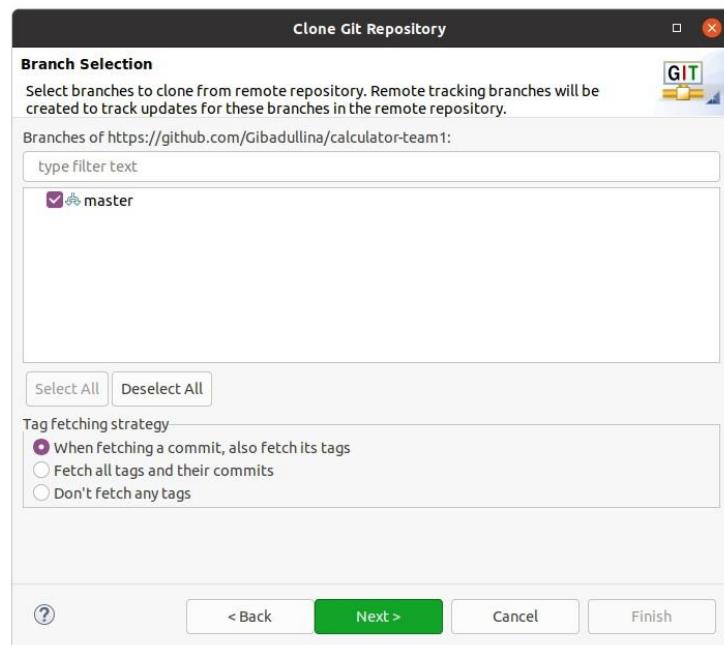


Рисунок 49 - Выбор ветки

Выбор места назначения (рис.50).

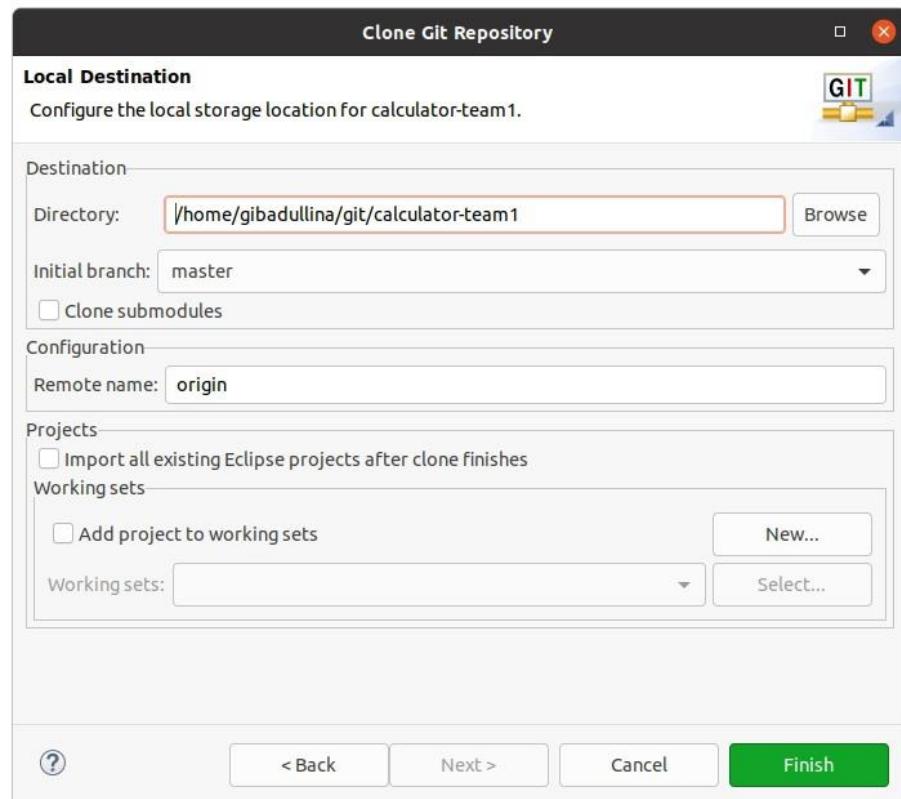


Рисунок 50 -Окно Local Destination

В итоге будет получен локальный репозиторий (рис.51).

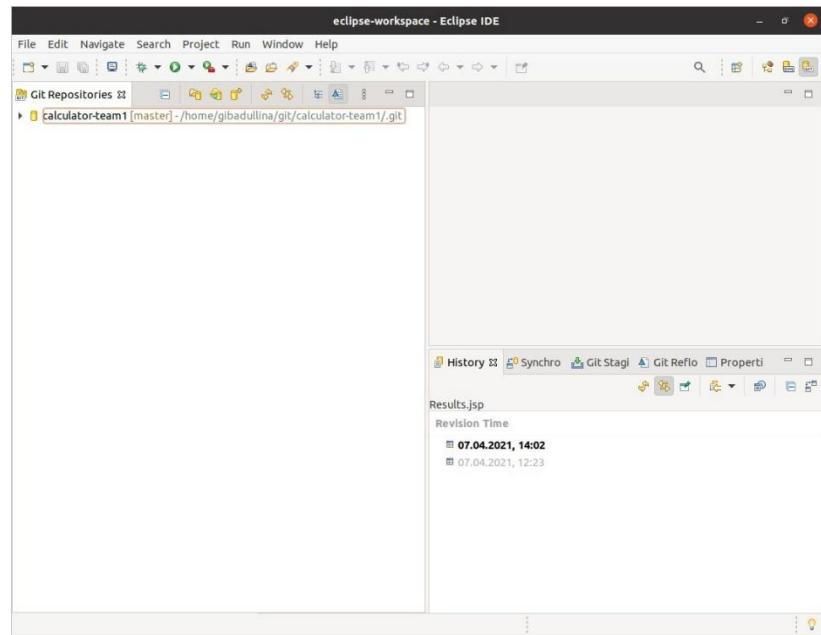


Рисунок 51 - Клонированный проект

Для импорта проекта (получение рабочей копии) необходимо перейти к «*Working Tree*», вызвать контекстное меню и выбрать «*Import projects...*» (рис.52).

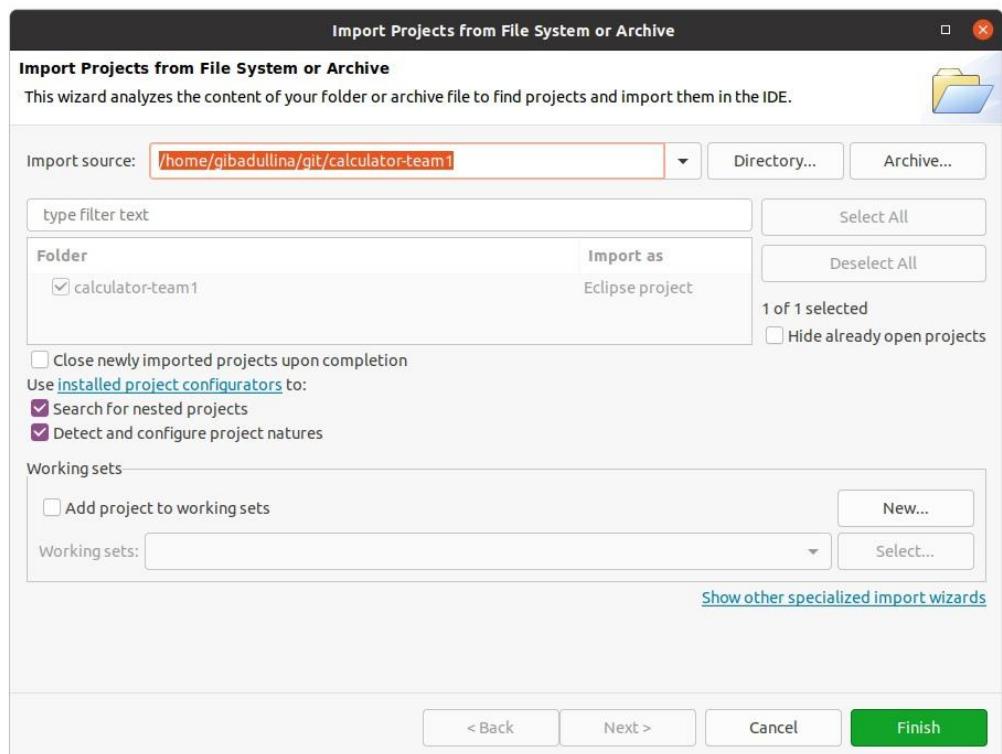


Рисунок 52 - Импорт проекта

Далее нажимается *Finish* и в *Project Explorer* появляется импортированный проект (рис.53).

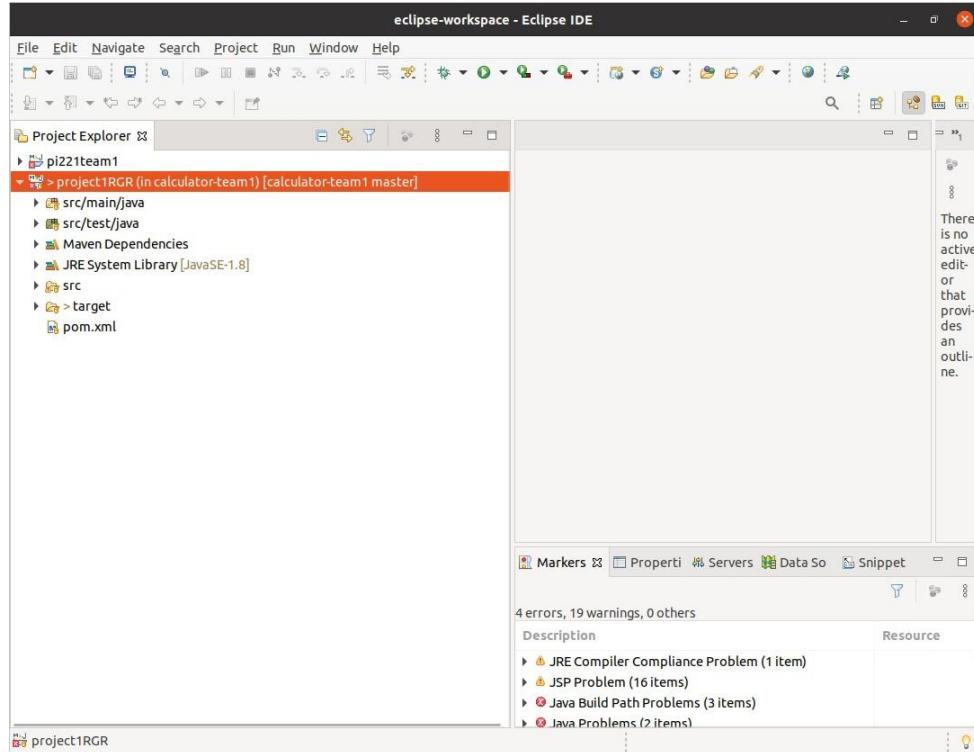


Рисунок 53 - Импортированный проект

## **Раздел 5. Реализация исходного кода по зонам ответственности**

В данном разделе представлена таблица с указанием зон ответственности (табл.1).

Таблица 1 - Зоны ответственности

<b>№</b>	<b>ФИО разработчика/модератора</b>	<b>Зона ответственности</b>
1	Гибадуллина Элина Юлаевна	Описание по зоне ответственности:auth.css, style.css, UtilHelper (интерфейс)
2	Газин Даниэль Раилевич	Описание по зоне ответственности: BaseSolver (абстрактный класс), Solver, RegisterController
3	Катасонов Серафим Александрович	Описание по зоне ответственности: AuthController, ExportController, auth.js, main.js
4	Рафиков Данил Римович	Описание по зоне ответственности: engine.jsp, index.jsp, класс Util, сервлет CalculatorController, EditController, ExitController, admin.js

Ссылка на репозиторий: <https://github.com/Gibadullina/calculator-team1>.

В ходе выполнения курсовой работы разрабатывается веб-приложение.

Веб-приложение (Web application) – это набор сервлетов, HTML страниц,

классов и других ресурсов, которые составляют законченное приложение на веб-сервере.

Программный код расположен в ПРИЛОЖЕНИИ 2.

Диаграмма классов (см. Раздел 1) являлась прототипом, поэтому внесены исправления под веб-приложение (рис.54).

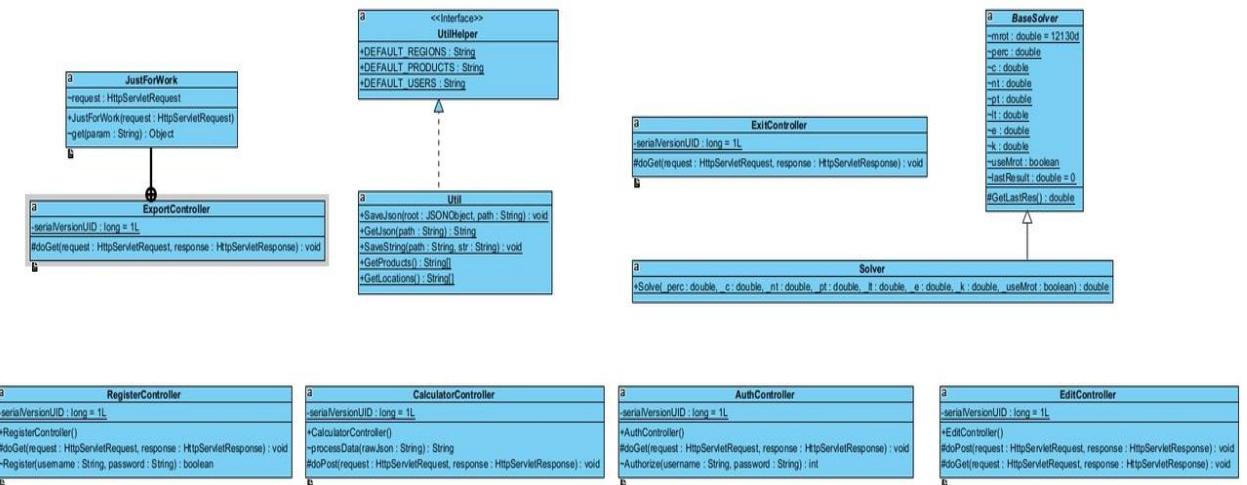


Рисунок 54 - Диаграмма классов

## **Раздел 6. Сборка и тестирование программного продукта**

В курсовой работе тестирование предполагает разработку и использование UNIT-тестов. Каждый участник Группы разрабатывает один UNIT-тест для своей зоны ответственности, используя фреймворк JUnit.

Описание UNIT-тестов и ответственные за их реализацию показаны в таблице 2.

Таблица 2 - Описание UNIT-тестов

<b>№</b>	<b>ФИО разработчика/модератора</b>	<b>Описание UNIT-теста</b>	<b>№ приложения</b>
1	Гибадуллина Элина Юлаевна	RegisterTest: тест проверяет создание пользователей с одинаковыми логинами; тест выполняется для класса <i>RegisterController</i> ; данные "admin", "admin"; ожидаемый результат: ложный результат.	См. Приложение 3
2	Газин Даниэль Раилевич	SolverTest: тест проверяет верно ли выполнены вычисления классом <i>Solver</i> ; тест выполняется для класса <i>Solver.java</i> ; данные <i>Solver(13, 15, 5, 1000, 10, 1000, 1.5f, true)</i> ; ожидаемый результат: равно 16007.999668121338	См. Приложение 4

## Продолжение таблицы 1

3	Катасонов Серафим Александрович	<i>AuthTets</i> : тест проверяет возможность авторизации под учетной записью администратора; тест выполняется для класса <i>AuthController</i> ; данные "admin", "admin"; ожидаемый результат: равенство 2	См. Приложение 5
4	Рафиков Данил Римович	<i>SaveJsonTest</i> : тест проверяет сохранения файла json; исходные данные название файла «test»; ожидаемый результат true	См. Приложение 6

### Инструкция по сборке проекта.

Описание *pom.xml* - Это специальный XML-файл, который всегда хранится в базовой директории проекта и называется *pom.xml*. Файл POM содержит информацию о проекте и различных деталях конфигурации, которые используются *Maven* для создания проекта:

#### Используемые плагины:

- *maven-compiler-plugin* - плагин для правильной компиляции;
- *maven-war-plugin* - плагин для создания war-файла;
- *maven-dependency-plugin* - плагин копирования зависимостей.

#### Используемые зависимости:

- *Junit 4.12* - зависимость для работы с JUnit
- *javax.servlet-api* - Java Servlet API (стандартизированный API, предназначенный для реализации на сервере и работе с клиентом по схеме запрос-ответ)
- *el-api* - язык API

- *json-simple* - набор инструментов Java для JSON
- *poi-ooxml* - для доступа к файлам .xls
- *jstl* - для работы с JSTL (стандартная библиотека тегов);
- *webapp-runner* - включение webapp-runner в путь к классам проекта.

### Описание *web.xml*:

Дескриптор развертывания, который описывает как нужно разворачивать на веб-сервере веб-приложение. В файле *web.xml* указывается по каким URL надо обращаться к приложению, указываются настройки безопасности. Путь к *webapp-runner.jar*.

### Структура проекта по каталогам:

Как видно на рисунке 55, проект calculator-team1 включает в себя каталоги *src/main/java*, *src/main/webapp* и *target*, а также файл *pom.xml*.

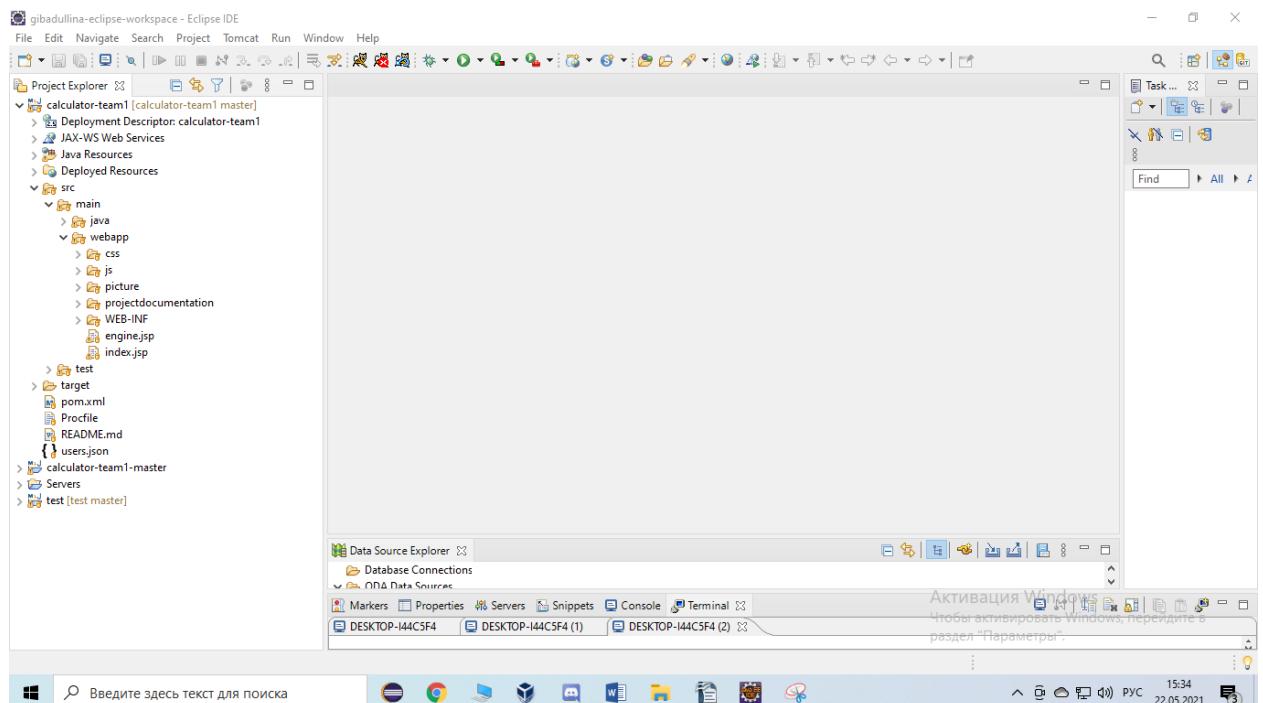


Рисунок 55 - Структура проекта-1

В свою очередь, каталог *webapp* включает в себя подкаталоги:

- *css* (содержит файлы *auth.css*, *style.css* - таблица стилей, определяющая позиционирование и отображение контента на веб-странице);

- *js* (содержит файлы *admin.js*, *auth.js*, *main.js*);
- *picture* (содержит *logo-aviastroy.jpg* - логотип предприятия ООО «Авиастрой», *icon.ico* - иконка, *icon64.png*);
- *projectdocumentation* (каталог с необходимой документацией программного продукта, включает файл *Specification.pdf* - ТЗ разрабатываемого ПО; файлы *UserManual.pdf* - Руководство пользователя; *ProgrammerManual.pdf* - Руководство программиста и *ExplanatoryNote.pdf* - ПЗ);
- WEB-INF (файл *web.xml*)

А также файлы:

- *engine.jsp* - JSP-файл для отображения страницы с Калькулятором;
- *index.jsp* - JSP-файл для отображения главной страницы;
- *Procfile* (в данном файле описываются имена процессов и команды, ассоциированные с ними);
- *README.md* (в данном файле содержится ссылка с *Travis*) - нужен для создания динамического бейджа.

Подкаталог *scr/main/java/com/skylabs* (рис.56) проекта содержит папки *baselogic* и *controllers*.

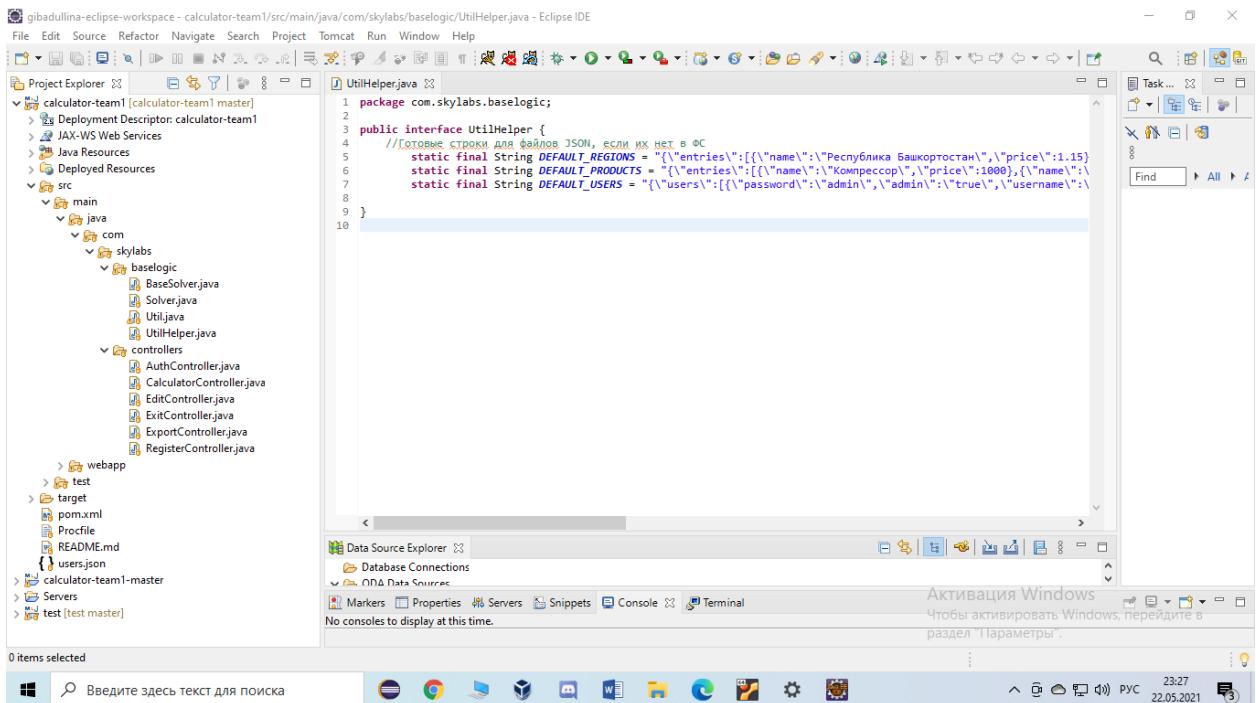


Рисунок 56 - Структура проекта-2

Папка *baselogic* включает следующие файлы:

- *BaseSolver.java* - абстрактный класс;
- *Solver.java* - класс с расчетом по формулам;
- *UtilHelper.java* - интерфейс;
- *Util.java* - класс для файла с настройками админа;

Папка *controllers* включает следующие файлы:

- *AuthController.java* - сервлет для авторизации пользователя;
- *CalculatorController.java* - сервлет проведения расчетов с входными данными;
- *EditController.java* - модуль администратора с редактированием;
- *ExitController.java* - выход
- *ExportController.java* - экспорт в печатную форму в формате .xls
- *RegisterController.java* - сервлет для регистрации пользователя.

Подкаталог *scr/test/java/com/skylabs* (рис.57) содержит папки, где находятся тесты для классов, *baselogic* (*SolverTest, SaveJsonTest*) и *controllers* (*AuthTest, RegisterTest*).

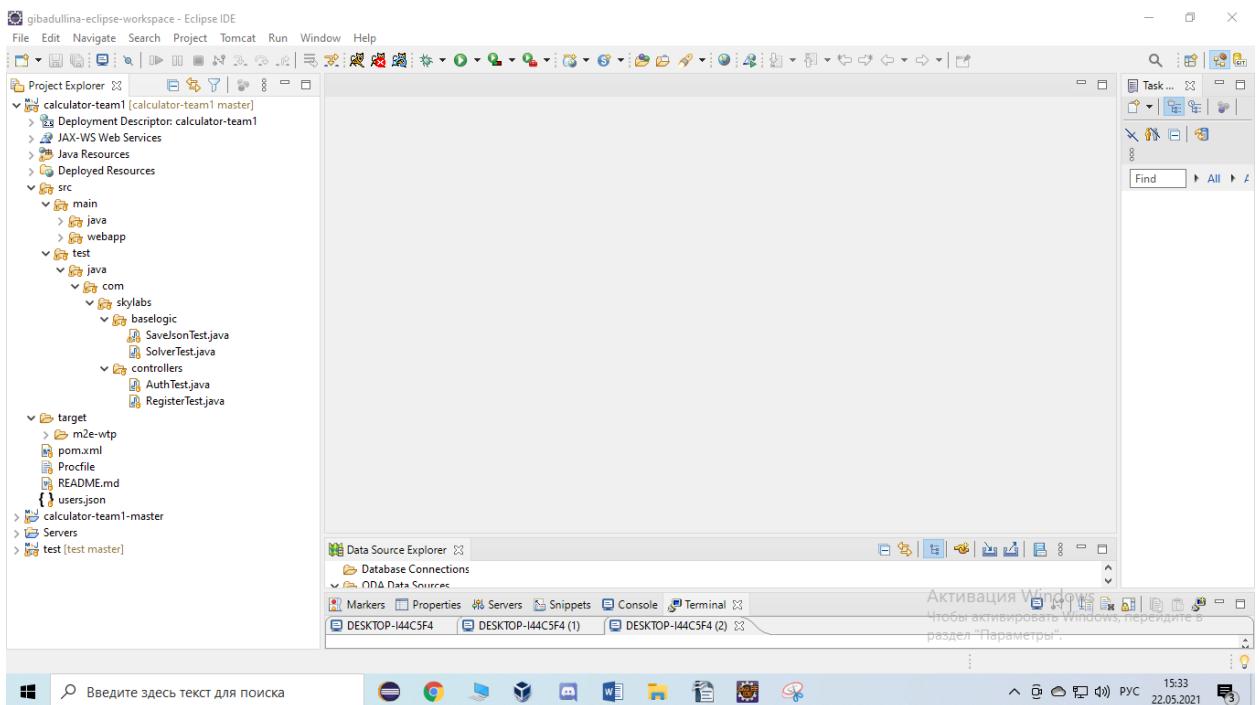


Рисунок 57 - Структура проекта-3

В папке target(рис.58) расположены подкаталог /m2e-wtp/web-resources/META-INF с файлом MANIFEST.MF.

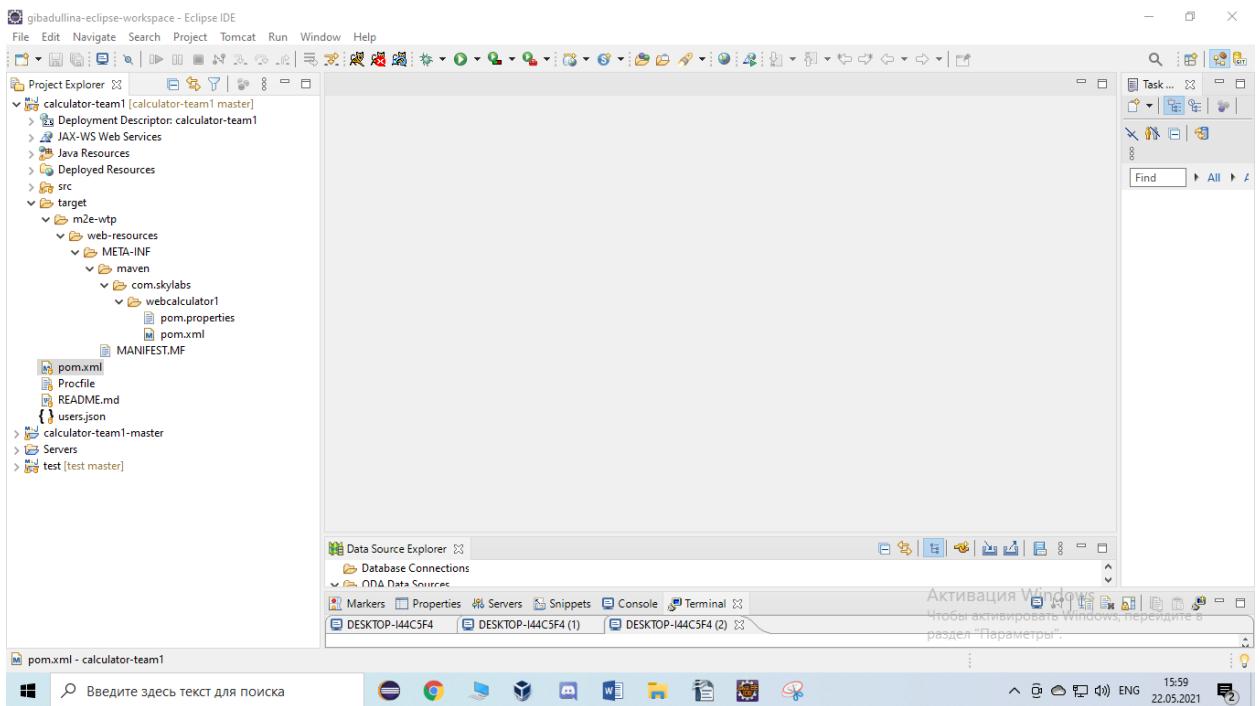


Рисунок 58 -Структура проекта -4

Процесс сборки проекта с основными действиями разработчика:

Для того, чтобы собрать Maven Project нужно выбрать пункты контекстного меню, как на рисунке 59, и команду «Maven Build».

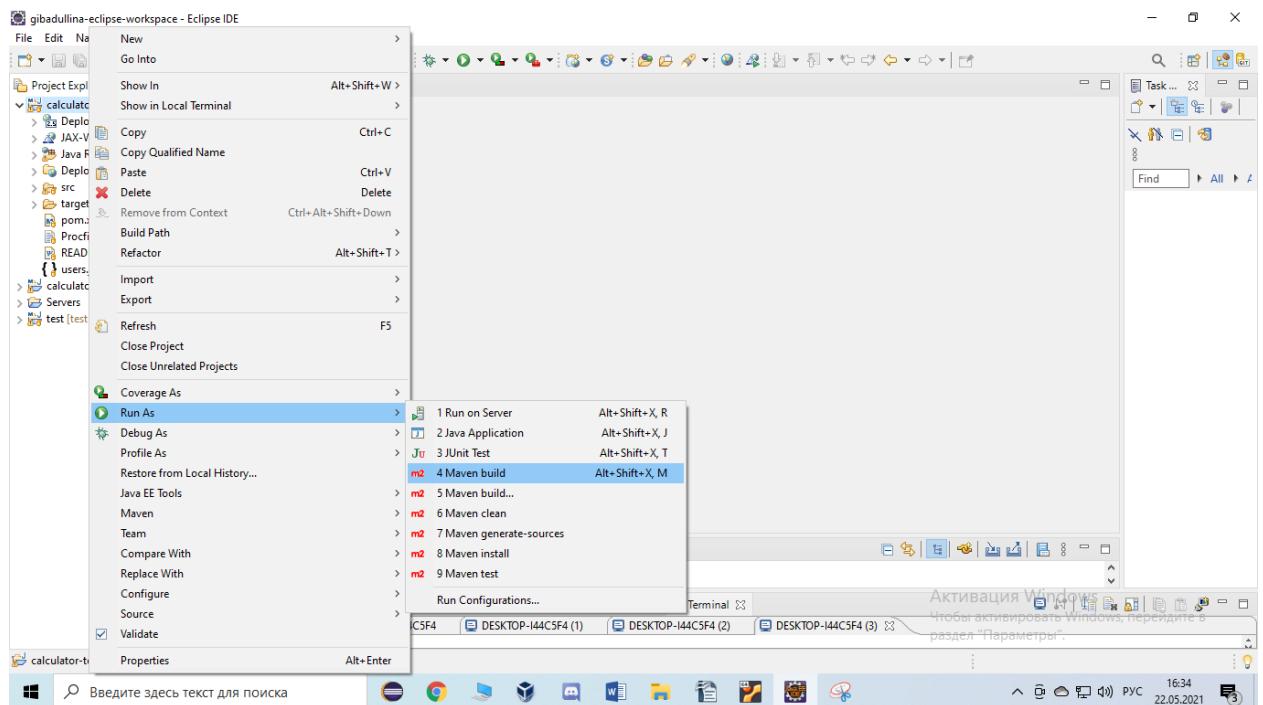


Рисунок 59 - Maven

В появившемся окне в поле goal вводим package. При успешной сборке будет написано «BUILD SUCCESS».

Собранный war-файл расположен в папке *artifacts*.

Данная инструкция поможет разработчику осуществить сборку проекта, однако в рамках данной курсовой работы сборка осуществляется через сервис *Travis CI* и настройку конфигурационного файла (см. Раздел 7).

## Раздел 7. Настройка программной среды для развертывания и запуска программного продукта

Непрерывная интеграция (*CI*, англ. Continuous Integration) – это практика разработки программного обеспечения (ПО), которая заключается в выполнении частых автоматизированных сборок проекта для скорейшего выявления и решения интеграционных проблем.

Для сборки и тестирования проекта используется распределенный веб сервис *Travis CI*.

Для сборки *Java* проекта в *Travis CI* необходимо создать в корне репозитория файл *.travis.yml*. Данный файл является конфигурационным и определяет цикл сборки.

Структура файла *.travis.yml* представлена на рисунке 60.

```
1 language: java
2 before_script:
3   - echo "Starting build"
4 script:
5   - mvn clean package
6 after_script:
7   - echo "Script finished"
8 after_success:
9   - echo "Build ready"
10  - ls -l $TRAVIS_BUILD_DIR/target
11 after_failure:
12  - echo "Build failure"
13 deploy:
14 skip_cleanup: true
```

Рисунок 60 - Структура файла *.travis.yml*

В конфигурации указывается язык программирования и далее указание шагов из полного цикла сборки *Travis CI*:

- 1) *before\_script* - выполнение команд до сборки;
- 2) *script* – выполнение сборки;
- 3) *after\_script* - выполнение команд после сборки;
- 4) *after\_success* - выполнение действий, когда сборка завершается успешно;
- 5) *after\_failure* - выполнение действий после неудачной сборки;
- 6) *deploy* - выполнение развертывания.

В курсовой работе подготовленной средой является платформа *Heroku*. Для работы созданного ранее веб-приложения необходимо сообщить *Heroku* как запустить приложение. Для этого *Heroku* требуется файл *Procfile* в корне папки приложения.

После того как приложение на *Heroku* создано, соединение *Heroku* с репозиторием *GitHub* установлено, конфигурационный файл *Procfile* создан и помещен в корне проекта, выполняем автоматическое развертывание. в разделе «*Deploy*» подключить репозиторий *GitHub*, а для автоматического развертывания необходимо поставить галочку в поле «*Wait for CI to pass before deploy*» и нажать на кнопку *Enable Automatic Deploys*.

Файлы работающей программы. В папке *artifacts* расположен *war*-файл, внутри которого описана структура каталогов (см. Раздел 6).

Условия работы программы. Веб-приложение работает в ОС (рис.61-63):

- Windows 10, а именно браузеры: Edge, Google Chrome, Mozilla Firefox;
- Linux Ubuntu - браузеры Mozilla Firefox, Google Chrome;
- Linux Mint - браузеры Mozilla Firefox, Google Chrome.

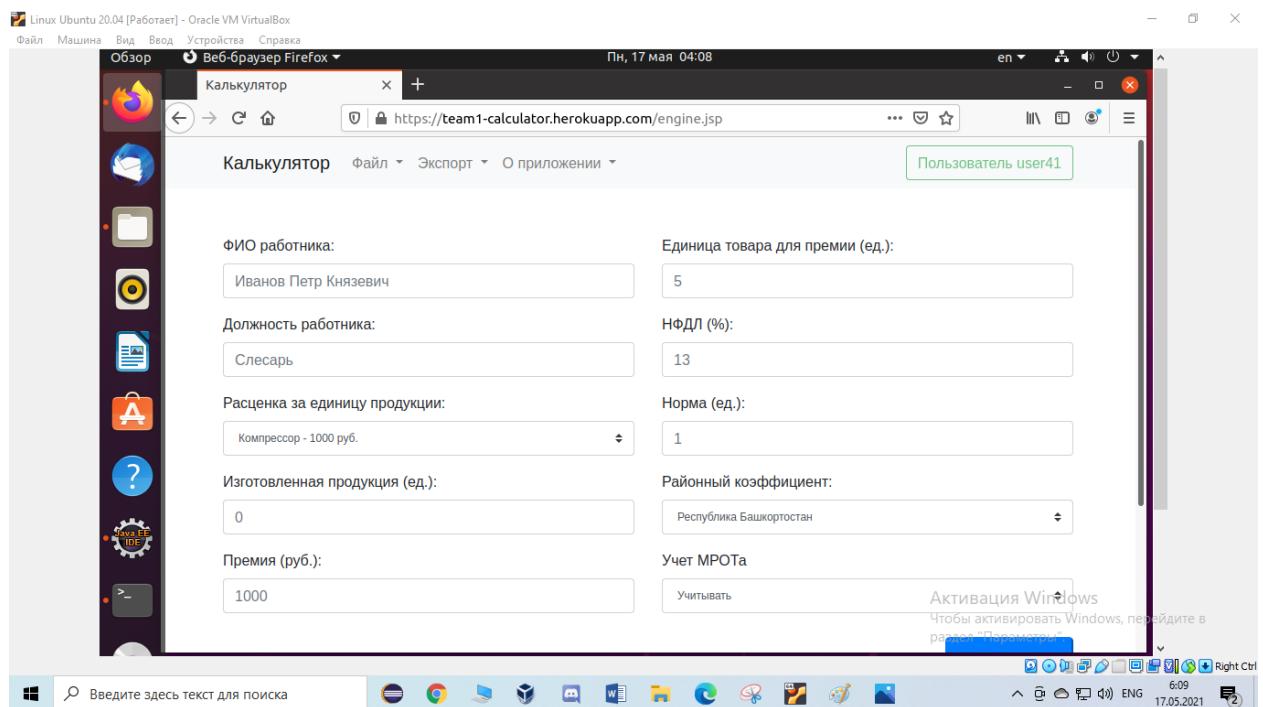


Рисунок 61 - Ubuntu 20.04 (Firefox)

team1-calculator.herokuapp.com/engine.jsp

Сервисы Авиабилеты Яндекс Главная | Кодовые... quegrande.org/aru... НОУ ИНТУИТ | Мо... Словарь Мультитр... Упражнение 3 уро... Список для чтения

Калькулятор Файл Экспорт О приложении Пользователь user41

ФИО работника:	Единица товара для премии (ед.):
Иванов Петр Князевич	5
Должность работника:	НФДЛ (%):
Слесарь	13
Расценка за единицу продукции:	Норма (ед.):
Компрессор - 1000 руб.	1
Изготовленная продукция (ед.):	Районный коэффициент:
0	Республика Башкортостан
Премия (руб.):	Учет МРОТа
1000	Учитывать
<b>Рассчитать</b>	

Рисунок 62 - Windows 10 (Google Chrome)

Linux Mint [Работает] - Oracle VM VirtualBox

Файл Машина Вид Ввод Устройства Справка

Калькулятор — Mozilla Firefox

Калькулятор Файл Экспорт О приложении Пользователь user41

ФИО работника:	Единица товара для премии (ед.):
Иванов Петр Князевич	5
Должность работника:	НФДЛ (%):
Слесарь	13
Расценка за единицу продукции:	Норма (ед.):
Компрессор - 1000 руб.	1
Изготовленная продукция (ед.):	Районный коэффициент:
0	Республика Башкортостан
Премия (руб.):	Учет МРОТа
1000	Учитывать
Активация Windows Чтобы активировать Windows, перейдите в раздел "Персональные".	

Рисунок 63 - Mint 20.1 (Firefox)

Ссылка на веб-приложение: <https://pi221team1-course.herokuapp.com/>.

## **Раздел 8. Руководство пользователя программного продукта**

При выполнении курсовой работы реализуется руководство пользователя (оператора). Данное руководство выполняется в соответствии с ГОСТ 19.505-79 «ЕСПД. Руководство оператора. Требования к содержанию и оформлению».

Согласно ГОСТ 19.505-79 руководство оператора должно содержать следующие разделы:

1. Назначение программы.
2. Условия выполнения программы.
3. Выполнение программы.
4. Сообщения оператору.

Разработанное руководство пользователя для программного продукта «Калькулятор сдельно-премиальной зарплаты» представлено в ПРИЛОЖЕНИИ 7.

## **Раздел 9. Руководство программиста программного продукта**

При выполнении курсовой работы реализуется руководство программиста. Данное руководство выполняется в соответствии с ГОСТ 19.504-79 «ЕСПД. Руководство программиста. Требования к содержанию и оформлению».

Согласно ГОСТ 19.504-79 руководство программиста должно содержать следующие разделы:

1. Назначение и условия применения программы;
2. Характеристики программы;
3. Обращение к программе;
4. Входные и выходные данные;
5. Сообщения.

Разработанное руководство программиста для программного продукта «Калькулятор сдельно-премиальной зарплаты» представлено в ПРИЛОЖЕНИИ 8.

## ЗАКЛЮЧЕНИЕ

В результате выполнения данного курсового проекта на тему «Разработка кроссплатформенного программного продукта на языке JAVA с использованием системы контроля версий» был разработан программный продукт «Калькулятор сдельно-премиальной зарплаты».

В ходе работы над проектом были выполнены задачи:

- Описана предметная область;
- Составлена документация к программному продукту: Техническое задание, Руководство пользователя (оператора), Руководство программиста;
- Настроена среда разработки для работы над проектом;
- Создано веб-приложение;
- Проведено тестирование компонентов программы;
- Произведена сборка и развертывание программного продукта.

Процесс выполнения курсовой работы был сопровожден изучением инструментального средства *Eclipse*, системы контроля версий *Git* совместно с сервисом *GitHub*, сервиса для сборки и тестирования программного обеспечения *Travis CI*, платформы для разработки и тестирования программных продуктов *Heroku*, системы управления проектами и задачами и оформлению документации.

# **ПРИЛОЖЕНИЕ 1**

## **ФГБОУ ВО УФИМСКИЙ ГОСУДАРСТВЕННЫЙ АВИАЦИОННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

СОГЛАСОВАНО  
Доцент кафедры АСУ ФГБОУ  
ВО УГАТУ  
Личная расшифровка  
подпись подписи  
27.03.2021

УТВЕРЖДАЮ  
Студент группы ПИ-221  
ФИРТ ФГБОУ УГАТУ, модератор  
Личная расшифровка  
подпись подписи  
26.03.2021

### **Калькулятор сдельно-премиальной зарплаты**

### **Техническое задание** **ЛИСТ УТВЕРЖДЕНИЯ** **1304.300001.000 ТЗ-ЛУ**

СОГЛАСОВАНО

Представитель  
команды разработчиков

Доцент кафедры АСУ  
ФГБОУ ВО УГАТУ  
Личная расшифровка  
подпись подписи  
27.03.2021

Студент группы ПИ-221  
ФИРТ ФГБОУ ВО УГАТУ, модератор  
Личная расшифровка  
подпись подписи  
26.03.2021

Изл. N	Подп.	Подп. И.дата

2021

Утвержден  
1304.300001.000 ТЗ-ЛУ

ФГБОУ ВО УФИМСКИЙ ГОСУДАРСТВЕННЫЙ АВИАЦИОННЫЙ  
ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

**Программный продукт  
«Калькулятор сдельно-премиальной зарплаты»  
Техническое задание**

**1304.300001.000 ТЗ**

**Листов 18**

№лз.№ помл.	Помл. №	В здам. №лз.№	№лз.№ куб.№	Помл. № куб.№

2021

## **СОДЕРЖАНИЕ**

Введение.....	4
1. Основание для разработки.....	5
2. Назначение разработки.....	6
3. Требования к программе или программному изделию.....	7
3.1. Требования к функциональным характеристикам.....	7
3.2. Требования к надежности.....	8
3.3. Условия эксплуатации.....	8
3.4. Требования к составу и параметрам технических средств.....	9
3.5. Требования к информационной и программной совместимости....	9
3.6. Требования к маркировке и упаковке.....	10
3.7. Требования к транспортированию и хранению.....	10
3.8. Специальные требования.....	10
4. Требования к программной документации.....	11
5. Технико-экономические показатели.....	12
5.1. Сравнительный анализ.....	12
6. Стадии и этапы разработки.....	14
7. Порядок контроля и приемки.....	15
Приложение П-1.....	17

## **ВВЕДЕНИЕ**

Техническое задание для программного продукта «Калькулятор сдельно-премиальной оплаты труда» разработан в рамках курсового проекта в соответствии с ГОСТ 19.201-78.

Темой курсового проекта является создание зарплатного калькулятора для сдельной формы оплаты труда.

Термины:

*Заработка плата* (оплата труда работника) - вознаграждение за труд в зависимости от квалификации работника, сложности, количества, качества и условий выполняемой работы, а также компенсационные выплаты (доплаты и надбавки компенсационного характера, в том числе за работу в условиях, отклоняющихся от нормальных, работу в особых климатических условиях и на территориях, подвергшихся радиоактивному загрязнению, и иные выплаты компенсационного характера) и стимулирующие выплаты (доплаты и надбавки стимулирующего характера, премии и иные поощрительные выплаты);

*Сдельная форма оплаты труда*: оплата производится за объем выполненных работ, независимо от потраченного времени.

*Сдельно-премиальная система оплаты труда*, наряду с оплатой по прямым сдельным расценкам, предусматривает премирование за перевыполнение нормы выработки и за достижение количественных и качественных показателей, определенных действующими условиями премирования.

В курсовой работе ведется разработка калькулятора заработной платы для ООО «Авиа-строй», которое занимается производством деталей реактивных двигателей.

## **1. Основания для разработки**

Основанием для разработки технического задания является выполнение курсовой работы по дисциплине «Информационные системы» в ФГБОУ ВО «УГАТУ» по направлению подготовки бакалавра 09.03.03 Прикладная информатика в экономике.

Документы на основании, которых ведется разработка: учебный план, методические указания.

Организация, утвердившая документ: Федеральное государственное бюджетное образовательное учреждение высшего образования «Уфимский государственный авиационный технический университет».

Наименование: программный продукт «Калькулятор сдельно-премиальной зарплаты».

## **2. Назначение разработки**

Программа будет использоваться двумя группами пользователей: простой пользователь и администратор.

Функциональное назначение программного продукта: программа предназначена для автоматизации расчета заработной платы для сотрудника-пользователя и возможность изменения настроек для администратора.

Эксплуатационное назначение: сотрудники, работающие в структурном подразделении «Цех», например, рабочий.

### **3. Требования к программе или программному изделию**

#### **3.1. Требования к функциональным характеристикам**

Для программы реализована многопользовательская работа. Поэтому в программном продукте есть модуль для авторизации пользователей, выделяющий две категории пользователей: простой пользователь и администратор.

Для авторизации необходимо заполнить два поля: логин и пароль.

Также у пользователей есть возможность регистрации.

Для простого пользователя предусмотрены следующие возможности:

- Ввести данные для расчета заработной платы;
- Получить результат расчёта;
- Сформировать печатную форму в формат *.xls* с результатами.

Для администратора есть возможность изменения настроек главной формы пользователя. Он способен изменять расценку на товары и значение районных коэффициентов.

Организация входных данных для пользователя. Пользователю для получения результатов расчёта необходимо заполнить следующие поля:

- ФИО работника;
- Должность работника;
- Изготовленная продукция, ед.;
- Премия, руб.;
- Единица товара для премии, ед.;
- НДФЛ, %;
- Норма, ед.

Расценка за единицу продукции и районный коэффициент выбирается из выпадающего списка.

Входные данные для администратора: расценка за единицу произведенной турбины и наименование региона-значение коэффициента.

Выходные данные для пользователя. Результат расчета отображается в текстовой поле, а также на форме. Есть возможность сформировать печатную форму в формате .xls, содержащую данные о пользователе и расчет.

Измененные настройки администратора хранятся в файле настройки.

Временные характеристики. «Калькулятор сдельно-премиальной зарплаты» должен режиме реального масштаба времени.

### **3.2. Требования к надежности**

Пользователю, работающему с программой через веб-браузер должен быть предоставлен непрерывный доступ к веб-приложению, расположенному по определённому url-адресу. Веб-сервис не должен непредвиденно прерывать свою работу.

Время восстановления после отказа в случае отказа работы серверной части и последующей недоступности веб-приложения не должно превышать одни рабочие сутки.

В случае отказа работы, не связанного с разрабатываемым программным обеспечением (вызванного неисправностью технических средств), время восстановления не должно превышать необходимого для исправления неисправностей.

Отказ программы вследствие некорректных действий оператора (пользователя) должен быть исключён.

### **3.3. Условия эксплуатации**

Требования к климатическим условиям эксплуатации не предъявляются.  
Обслуживание не требуется.

При настройке системы необходим администратор. Эксплуатация производится пользователем. Для работы с приложением требования по квалификации пользователя и администратора отсутствуют.

### **3.4. Требования к составу и параметрам технических средств**

Технические требования к серверу, необходимые для функционирования веб-приложения:

- процессор, ОЗУ и видеокарта (интегрированная или внешняя) должны позволять запустить операционную систему *Windows*, *Linux Ubuntu* или *Linux Mint*;
- Поддерживаемые протоколы передачи данных: HTTP / HTTPS;
- Количество подключенных пользователей 250;
- Процессор 4 ядра, тактовая частота 2.90 ГГц и выше;
- Платформа 32-х или 64-х разрядная;
- Оперативная память- 10 ГБ и выше;
- Жесткий диск- 300 МБ свободного объема и выше.
- Поддержка языка программирования Java, поддержка хранения и обработки данных;
- Интеграция с github;
- Поддержка Apache http.

### **3.5. Требования к информационной и программной совместимости**

Серверная часть программы должна быть написана на языке *Java*. Клиентская сторона на *HTML*, *CSS*, *JavaScript* с применением дополнительных библиотек.

Для совместной разработки программного продукта будет использовано инструментальное средство *Eclipse*.

Кроссплатформенность будет предполагать, что программный продукт будет работать в таких операционных системах, как: *Windows*, *Linux Ubuntu* и *Linux Mint*.

### **3.6. Требования к маркировке и упаковке**

Программа поставляется в виде war-файла.

### **3.7. Требования к транспортированию и хранению**

Проект будет расположен на веб-сервисе *github.com*. Развёртывание программного продукта на платформе *Heroku*.

### **3.8. Специальные требования.**

Программа должна обеспечивать взаимодействие с пользователем посредством интуитивно-понятного графического пользовательского интерфейса.

#### **4. Требования к программной документации**

Предварительный состав документации:

- 1) Техническое задание (настоящий документ). Разработка согласно ГОСТ 19.201-78. «Единая система программной документации (ЕСПД). Техническое задание. Требования к содержанию и оформлению» в соответствии с ГОСТ 19.106-78 и ГОСТ 2.301-68. Лист утверждения и титульный лист оформляют в соответствии с ГОСТ 19.104-78.
- 2) Руководство оператора (пользователя). Разработка регламентируется стандартом ГОСТ 19.505-79.
- 3) Программа и методика испытаний. Разработка регламентируется стандартом ГОСТ 19.301-79.
- 4) Текст программы. Разработка регламентируется стандартом ГОСТ 19.401-78.
- 5) Руководство программиста. Разработка регламентируется стандартом ГОСТ 19.504-79 «ЕСПД. Руководство программиста. Требования к содержанию и оформлению»
- 6) Пояснительная записка оформляется в соответствии с требованиями ГОСТ 19.106-78 «ЕСПД. Виды программ и программных документов», ГОСТ 19.201-78 «ЕСПД. Техническое задание. Требования к содержанию и оформлению», ГОСТ 19.404-79 «ЕСПД. Пояснительная записка. Требования к содержанию и оформлению» и ГОСТ 19.701-90 «ЕСПД. Схемы алгоритмов, программ, данных и систем. Обозначения условные и правила выполнения».

## 5. Технико-экономические показатели

Данный программный продукт позволяет сократить временные затраты на расчет сдельной заработной платы сотрудников, работающих в структурном подразделении «Цех».

Рост потребности не ожидается, так как приложение заточено строго для работы в ООО «Авиа-строй».

В данный калькулятор уже заложены базовые данные, которые могут потребоваться при работе ООО «Авиа-строй», в то время как программное обеспечение конкурентов необходимо настраивать.

### 5.1. Сравнительный анализ

Сравнительный анализ между разрабатываемым программным продуктом «Калькулятор сдельно-премиальной зарплаты» и ПО, которое есть на рынке.

Преимущества веб-приложения перед калькулятором «Расчет сдельной зарплаты» <https://pravo.team/trudovoe/oplata/sdelno-premialnaa.html>:

- Возможность авторизации;
- Возможность сохранения результатов расчета;
- Вывод печатной формы с результатами;
- Расширенная функциональность: кроме единиц продукции и расценки, наш пользователь может вводить свои личные данные, а также учитывать такие значения при расчете, как МРОТ, районный коэффициент, НДФЛ.

Программа БухСофт для расчета зарплаты:

<https://www.buhsoft.ru/news/5682-sud-ne-razreshil-otkazyvat-nalogovikam-esli-oni-ne-ukazali-rekvizity-v-trebovaniy>. Эта программа способна рассчитать автоматически зарплату по сдельной форме оплаты труда, а также сделать это для всех работников одновременно. Конечно, это является преимуществом перед программным продуктом, разрабатываемым в рамках курсовой работы.

Однако важными отличием, которые являются достоинством:

- Понятно-интуитивный интерфейс, который достаточно прост и не требует того, чтобы разбираться, как им пользоваться.
- Основное преимущество - программный продукт «Калькулятор сдельно-премиальной зарплаты» является бесплатным, в отличие от БухСофта.

Рассмотрим зарубежные аналоги программного продукта.

Приложение App Store для расчета сдельной зарплаты:<https://apps.apple.com/ru/app/pieceseray-piecework-calculator/id1401931208>. Преимущества «Калькулятора сдельно-премиальной зарплаты»:

- Подстройка под стандарты Российской Федерации;
- Бесплатное использование;
- Корреспонденция: работа программы в ОС Windows и Linux;
- Возможность формирования печатной формы.

Также в приложении PiecePay: Piecework Calculator пользователь не может вводить свои личные данные.

Piece Rate Calculator: <https://www.dol.gov/agencies/whd/workers-with-disabilities/section-14c/calculators/PieceRate>. Преимущества перед этим калькулятором:

- Возможность авторизации;
- Возможность сохранения результатов расчета;
- Подстройка под стандарты Российской Федерации;
- Расширенная функциональность: ввод личных данных, учет налога.

Вывод по анализу: программный продукт, разрабатываемый в рамках курсового проекта, обладает рядом преимуществ перед аналогами. Он совмещает в себе различные возможности и при этом является бесплатным и доступным.

## **6. Стадии и этапы разработки**

В данном подразделе технического задания установлены необходимые стадии разработки, этапы и содержание работ, а также сроки предъявления.

Для выполнения курсовой работы установлен срок - 12 учебных недель. Этапы и стадии разработки представлены в виде план-графика выполнения курсовой работы (табл. 1).

Таблица - 1 План-график выполнения курсовой работы

Наименование этапа работ	Трудоемкость выполнения, час	Процент к общей трудоемкости выполнения	Срок предъявления консультанту
Получение и согласование задания	1,7	1,7%	27 неделя
Раздел 1. Описание предметной области	20	20%	29 неделя
Раздел 2. Техническое задание на создание программного продукта	10	10%	30 неделя
Раздел 3. Настройка среды разработки для операционных систем семейств Windows и Linux	10	10%	31 неделя
Раздел 4. Настройка среды разработки для подключения к системе контроля версий	7	7%	32 неделя
Раздел 5. Реализация исходного кода по зонам ответственности	23	23%	34 неделя
Раздел 6. Сборка и тестирование программного продукта	8	8%	35 неделя

Продолжение таблицы 1

Раздел 7. Настройка программной среды для развертывания и запуска программного продукта	10	10%	36 неделя
Раздел 8. Руководство пользователя программного продукта	10	10%	37 неделя
Защита	0,3	0,3%	38 неделя
<b>Итого:</b>	<b>100</b>	<b>100</b>	

## **7. Порядок контроля и приемки**

В данном разделе технического задания указаны виды испытаний и общие требования к приемке работы. То есть требования, по которым будет приниматься данная курсовая работа и проводиться защита.

Требования к разрабатываемому кроссплатформенному программного продукту:

1. Кроссплатформенность – способность программного продукта работать с несколькими аппаратными платформами или операционными системами.
2. Модульность – программный продукт состоит из частей – модулей, которые можно независимо друг от друга программирования, транслировать, отлаживать (проверять, исправлять).
3. Поддержка коллективной работы с исходным текстом и документацией по программному продукту.
4. Наличие интуитивно-понятного графического интерфейса для пользователей.
5. Применение и использованием файл-серверной, клиент-серверной, либо иной технологии реализующей многопользовательскую работу. Необходимость реализации модуля авторизации для пользователей программного продукта, где выделяется как минимум две категории пользователей: администратор и простой пользователь.
6. Входной информацией для программного продукта является необходимая информация для решения поставленной задачи, где данную информацию вводит пользователь по средствам устройств ввода информации.
7. Выходная информация отображается на форме и нужно предусмотреть формирование как минимум одной печатной формы (*.doc, .docx, .odt, .pdf, .xls*).

Для допуска к защите курсовой работы необходимо загрузить пояснительную записку на <https://sdo.ugatu.su/> в соответствующий раздел курса «Информационные системы».

Защита курсовой работы проводится на 38 учебной неделе в соответствии с планом-графиком.

Защита курсовой работы позволяет определить теоретический уровень подготовки студента; умение работать со средой *Eclipse*, веб-сервисом *GitHub*, сервисом *Travis CI* и платформой *Heroku*; формулировать выводы по полученным результатам.

Процедура защиты курсовой работы предполагает следующие этапы:

1. Настройка среды *Eclipse* в нескольких операционных системах разных семейств.
2. Клонирование репозитория *GitHub*, извлечение рабочей копии и выполнение основных команд.
3. Работа с сервисом *Travis CI*.
4. Выполнить развертывание и запуск программного продукта.
5. Знание своей зоны ответственности.

## ПРИЛОЖЕНИЕ П-1

### ДОПОЛНИТЕЛЬНЫЕ СВЕДЕНИЯ

В ходе выполнения курсовой работы существует необходимость использования инструментальных средств, СКВ, серверов и т.д.

Для совместной разработки программного продукта будет использовано инструментальное средство *Eclipse*. *Eclipse* – свободная среда разработки модульных кроссплатформенных приложений.

В качестве системы контроля версий (СКВ) использован *Git* для поддержки командной работы с исходным текстом и документацией по программному продукту. *Git* – это распределенная система управления версиями. Для доступа к удаленному репозиторию используется веб-сервис *github*. Веб-сервис основан на системе контроля версий *Git* и разработан на *Ruby on Rails* и *Erlang* компанией *GitHub*.

Для непрерывной интеграции использован сервис для сборки и тестирования программного обеспечения *Travis CI*. В качестве платформы непрерывной интеграции *Travis CI* поддерживает процесс разработки, автоматически тестируя программный код, обеспечивая обратную связь. Также *Travis CI* может автоматизировать другие процессы разработки, управляя развертыванием и уведомлениями.

Для развертывания программного продукта существует необходимость использования платформы *Heroku*. *Heroku* – это облачная платформа, основанная на управляемой контейнерной системе, с интегрированными службами передачи данных и мощной экосистемой для развертывания и запуска современных приложений.

Разрабатываемое веб-приложение должно быть ориентировано на клиент-серверную архитектуру. Под клиентом будет пониматься веб-браузер пользователя, через который идет обращение к серверу. В качестве сервера будет выступать платформа *Heroku*.

Кроссплатформенность предполагает, что программный продукт имеет способность работать с несколькими операционными системами. Поэтому существует необходимость того, что работа программного продукта будет производиться в ОС *Windows*, *Linux Ubuntu* и *Linux Mint*.

## **ПРИЛОЖЕНИЕ 2**

### *ПРОГРАММНЫЙ КОД*

#### *BASESOLVER.JAVA*

```
package com.skylabs.baselogic;

public abstract class BaseSolver {
    static final double mrot = 12130d;
    static double perc,c,nt,pt,lt,e,k;
    static boolean useMrot;
    static double lastResult = 0;

    //возвращаем последнее расчитанное значение
    protected static double GetLastRes()
    {
        return lastResult;
    }
}
```

## *SOLVER.JAVA*

```
package com.skylabs.baselogic;

public class Solver extends BaseSolver{

    //Расчёт по формуле
    public static double Solve(double _perc, double _c, double _nt,
    double _pt, double _lt, double _e, double _k, boolean _useMrot)
    {
        perc = _perc; // налог
        c = _c; // количество деталей
        nt = _nt; // количество деталей на получение полной премии
        pt = _pt; // премия
        lt = _lt; // норма
        e = _e; // цена детали
        k = _k; // районный коэф
        useMrot = _useMrot; //используется ли мрот

        double result = 0;
        double m = 0;
        if (nt > 0 && lt < c) { //если параметры премии заданы и
было произведено деталей свыше нормы
            m = e*c + (c-lt)/nt*pt;
        }
        else { // иначе считаем без учета премии
            m = e*c;
        }
        if (_useMrot && m < mrot) { //Учет МРОТ
            result = mrot*k*(1-perc/100);
        } else {
            result = m*k*(1-perc/100);
        }

        lastResult = result;
        return result;
    }
}
```

## *UTIL.JAVA*

```
package com.skylabs.baselogic;

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.FileWriter;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.io.Writer;
import java.nio.file.Path;

import org.json.simple.JSONArray;
import org.json.simple.JSONObject;
import org.json.simple.parser.JSONParser;

public class Util { //Вспомогательный класс для работы с ИО

    //Готовые строки для файлов JSON, если их нет в ФС

    static final String DEFAULT_REGIONS =
"{"entries": [{"name": "Республика Башкортостан", "price": 1.15}, {"name": "Ростовская область", "price": 1.1}, {"name": "Республика Дагестан", "price": 1.12}, {"name": "Республика Калмыкия", "price": 1.3}, {"name": "Ставропольский край", "price": 1.15}]}";

    static final String DEFAULT_PRODUCTS =
"{"entries": [{"name": "Компрессор", "price": 1000}, {"name": "Вентилятор", "price": 300}, {"name": "Турбина", "price": 400}, {"name": "Сопло", "price": 100}, {"name": "Смеситель", "price": 200}]}";

    static final String DEFAULT_USERS =
"{"users": [{"password": "admin", "admin": true, "username": "admin"}]}";
```

```

//Сохраняем корневой узел JSON в файл по указанному пути
public static void SaveJson(JSONObject root, String path) {
    try {
        FileWriter file = new FileWriter(path);
        file.write(root.toJSONString());
        file.flush();
        file.close();
    } catch (Exception ex) {
        ex.printStackTrace();
    }
}

//Получаем строку в виде сырого JSON по указанному пути
public static String GetJson(String path) {
    String json = "";
    try {
        File f = new File(path);
        if (!f.exists()) {
            if (path.equals("users.json"))
SaveString("users.json", DEFAULT_USERS);
            else if (path.equals("data.json"))
SaveString("data.json", DEFAULT_PRODUCTS);
            else if (path.equals("data_regions.json"))
SaveString("data_regions.json", DEFAULT_REGIONS);
        }
        try { //Считывание данных
            @SuppressWarnings("resource")
            BufferedReader buff = new BufferedReader(new
InputStreamReader(new FileInputStream(path), "UTF-8"));
            String line = "";
            while((line = buff.readLine()) != null) {
                json += line;
            }
        }
    }
}

```

```

        catch (Exception ex) {
            ex.printStackTrace();
            return "Internal error";
        }

        return json;
    }

    catch (Exception ex) {
        ex.printStackTrace();
        return null;
    }
}

//Сохранение строки в файл по указанному пути
public static void SaveString(String path, String str)
throws IOException {
    Writer out = new BufferedWriter(new OutputStreamWriter(
        new FileOutputStream(path), "UTF-8"));
    try {
        out.write(str);
    } finally {
        out.close();
    }
}

//Получение списка продуктов
public static String[] GetProducts() {
    try {
        String[] entries;
        JSONParser parser = new JSONParser();
        JSONObject root = (JSONObject)
parser.parse(Util.GetJson("data.json"));
        JSONArray users = (JSONArray) root.get("entries");

```

```

        entries = new String[users.size()]; //Наши
    продукты

        int i = 0;
        for(Object entry : users) {
            JSONObject entryy = (JSONObject) entry;
            entries[i] = entryy.get("name") + " - " +
entryy.get("price") + " руб.";
            i++;
        }

        return entries;
    }
    catch(Exception ex) {
        ex.printStackTrace();
    }
    return null;
}

//Получение коэффициентов по регионам
public static String[] GetLocations() {
    try {
        String[] entries;
        JSONParser parser = new JSONParser();
        JSONObject root = (JSONObject)
parser.parse(Util.GetJson("data_regions.json"));
        JSONArray users = (JSONArray) root.get("entries");

        entries = new String[users.size()]; //Наши
    региональные
        коэффициенты

        int i = 0;
        for(Object entry : users) {
            JSONObject entryy = (JSONObject) entry;
            entries[i] = entryy.get("name") + "";

```

```
    i++;
}

return entries;
}

catch (Exception ex) {
    ex.printStackTrace();
}

return null;
}

}
```

## AUTHCONTROLLER.JAVA

```
package com.skylabs.controllers;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.json.simple.JSONArray;
import org.json.simple.JSONObject;
import org.json.simple.parser.JSONParser;
import org.json.simple.parser.ParseException;
import com.skylabs.baselogic.Util;

//Сервлет для аутентификации
@WebServlet("/AuthController")
public class AuthController extends HttpServlet {
    private static final long serialVersionUID = 1L;

    public AuthController() {
        super();
    }

    //Процесс авторизации
    protected void doGet(HttpServletRequest request,
    HttpServletResponse response) throws ServletException,
    IOException {
        request.setCharacterEncoding("UTF-8"); //Устанавливаем
        UTF-8 кодировку, чтобы было все в порядке с кириллицей
        response.setCharacterEncoding("UTF-8");
    }
}
```

```

response.setContentType("text/html");

String username = request.getParameter("username");
//Считываем параметры из запроса

String password = request.getParameter("password");

int check = Authorize(username, password);
//Авторизуемся

if (check != 0) { //Если авторизация успешна
    request.getSession().setMaxInactiveInterval(-1);
//Устанавливаем куки через сессию, что пользователь
авторизован

    request.getSession().setAttribute("login",
"true");

    if (check == 2)
request.getSession().setAttribute("admin", "true"); //Проверяем,
не админ ли пользователь

    else request.getSession().setAttribute("admin",
"false");

PrintWriter writer = response.getWriter(); //Пишем
результат выполнения пользователю

try {
    request.getSession().setAttribute("login", true);
    request.getSession().setAttribute("username",
username);

    writer.println("1");
} finally {
    writer.close();
}

}

else { //В случае неудачи выдаем соответствующее
предупреждение

PrintWriter writer = response.getWriter();
try {

    writer.println("Неправильный логин или пароль!");
}

```

```

        request.getSession().setAttribute("login", false);
        request.getSession().setAttribute("username", "");
    } finally {
        writer.close();
    }
}

//Авторизация. 0 - неуспех, 1 - пользователь, 2 - админ
int Authorize(String username, String password) {
    String json = Util.GetJson("users.json"); //Получаем
JSON с пользователями

    try {
        JSONParser parser = new JSONParser();
        JSONObject root = (JSONObject) parser.parse(json);
//Парсим данные в работоспособный формат

        JSONArray users = (JSONArray) root.get("users");

        for(Object entry : users) { //Перебираем каждого
            JSONObject user = (JSONObject) entry;
            if (user.get("username").equals(username) &&
user.get("password").equals(password)) { //Если логин и пароль
совпали, успех
                if
((boolean)user.get("admin")).equals("true")) return 2; //В
случае, если пользователь является админом, возвращаем иной
результат
                else return 1; {}
            }
        }
    } catch (ParseException e) {
        e.printStackTrace();
        return 0;
    }
    return 0;
}
}

```

## **CALCULATORCONTROLLER.JAVA**

```
package com.skylabs.controllers;

import com.skylabs.baselogic.*;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.PrintWriter;
import org.json.simple.*;
import org.json.simple.parser.JSONParser;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

//Сервлет для расчётов
@WebServlet("/CalculatorController")
public class CalculatorController extends HttpServlet {
    private static final long serialVersionUID = 1L;

    public CalculatorController() {
        super();
        // TODO Auto-generated constructor stub

        //Процесс обработки данных
        String processData(String rawJson) {
            try {
                JSONParser parser = new JSONParser();
                JSONObject json = (JSONObject)parser.parse(rawJson);
                //Парсим данные в нужный формат
            }
        }
    }
}
```

```

//Base logic

double perc, c, nt, pt, lt, e = 0, k = 0;

//Последовательно берем данные из JSON

perc =
Double.parseDouble(String.valueOf(json.get("ndfl")));

pt =
Double.parseDouble(String.valueOf(json.get("prize")));

c = (long)json.get("production");

nt = (long)json.get("count");

lt = (long)json.get("normal");

long indexProduct = (long)json.get("val");

long indexCoeff = (long)json.get("location");

boolean useMrot = (boolean)json.get("mrot");



//Здесь получаем коэффициенты по продукции и регионам
из data.json и data_regions.json посредством дополнительных
парсингов

try {

parser = new JSONParser();

JSONObject root = (JSONObject)
parser.parse(Util.GetJson("data.json"));

JSONArray users = (JSONArray)
root.get("entries");

e =
Double.parseDouble(((JSONObject)users.get((int)indexProduct)).ge-
t("price").toString());

}

catch(Exception ex) {

ex.printStackTrace();


try {

parser = new JSONParser();

JSONObject root = (JSONObject)
parser.parse(Util.GetJson("data_regions.json"));

JSONArray users = (JSONArray)
root.get("entries");

```

```

        k =
Double.parseDouble(((JSONObject)users.get((int)indexCoeff)).get(
"price").toString());
    }

    catch (Exception ex) {
        ex.printStackTrace();
    }

    double result = Solver.Solve(perc, c, nt, pt, lt, e, k,
useMrot); //Делаем расчёт

    return "Заработка плата " + json.get("fio") + "
составит " + result + " у.е.";
}

catch (Exception ex) {
    return ex.getMessage();
}
}

//Обработка запроса POST

protected void doPost(HttpServletRequest request,
HttpServletResponse response) throws ServletException,
IOException {
    request.setCharacterEncoding("UTF-8"); //Нормализуем
кодировку
    response.setContentType("text/plain");
    response.setCharacterEncoding("UTF-8");

    BufferedReader reader = request.getReader();
    String result = "";
    while(true) {
        String line = reader.readLine(); //Берем данные из
тела запроса
        if (line != null) {
            result += line;
        } else break;
    }
}

```

```
try {
    result = processData(result); //Высчитываем
заработную плату
}
catch(Exception ex) {
    result = ex.getMessage();
}
PrintWriter writer = response.getWriter();
try {
    writer.println(result); //Отправляем результат
}
finally {
    writer.close();
} } }
```

## *EDITCONTROLLER.JAVA*

```
package com.skylabs.controllers;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import com.skylabs.baselogic.Util;

//Сервлет для редактирования коэффициентов регионов и продукции
public class EditController extends HttpServlet {

    private static final long serialVersionUID = 1L;

    public EditController() {
        super();
        // TODO Auto-generated constructor stub }

    //Обработка запроса, когда пользователь внес данные
    protected void doPost(HttpServletRequest request,
                          HttpServletResponse response) throws ServletException,
                          IOException {
        request.setCharacterEncoding("UTF-8");
        response.setCharacterEncoding("UTF-8");
        response.setContentType("text/html");

        BufferedReader reader = request.getReader();
        String rawJson = "";

        while(true) {
```

```

        String line = reader.readLine(); //Считываем из
тела запроса данные

        if (line != null) {
            rawJson += line;
        } else break;

    }

    if (request.getParameter("type").equals("products")) {
//Если редактировалась продукция, то сохраняем ее новые
коэффициенты

        Util.SaveString("data.json", rawJson);

    }

    else if
(request.getParameter("type").equals("regions")) { //Или
сохраняем новые данные по регионам

        Util.SaveString("data_regions.json", rawJson);

    }

}

//Если пользователь хочет получить список региональных
коэффициентов и продукции перед редактированием

protected void doGet(HttpServletRequest request,
HttpServletResponse response) throws ServletException,
IOException {

    request.setCharacterEncoding("UTF-8");
    response.setCharacterEncoding("UTF-8");
    response.setContentType("text/html");

    String json = "";
    if (request.getParameter("type").equals("products")) {
//Если продукция, то берем данные о ней

        json = Util.GetJson("data.json");

    } else if
(request.getParameter("type").equals("regions")) { //Или
региональные коэффициенты

        json = Util.GetJson("data_regions.json");

    }

}

```

```
PrintWriter writer = response.getWriter();
writer.println(json); //Пишем результат пользователю
}

}
```

## *EXITCONTROLLER.JAVA*

```
package com.skylabs.controllers;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

//Сервлет для выхода из сессии
public class ExitController extends HttpServlet {
    private static final long serialVersionUID = 1L;

    //Обработка запроса выхода из сессии
    protected void doGet(HttpServletRequest request,
                         HttpServletResponse response) throws ServletException,
                         IOException {
        request.setCharacterEncoding("UTF-8");
        response.setCharacterEncoding("UTF-8");
        response.setContentType("text/html");

        //Стираем куки в сессии
        request.getSession().setAttribute("login", "false");
        request.getSession().setAttribute("admin", "false");
        request.getSession().setAttribute("username", "");

        PrintWriter writer = response.getWriter();
        try {
            writer.println("1"); //Пишем о том, что все прошло
            успешно
        } finally {
            writer.close();
        }
    }
}
```

## *EXPORTCONTROLLER.JAVA*

```
package com.skylabs.controllers;

import java.io.IOException;
import java.io.OutputStream;
import java.text.DecimalFormat;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.apache.poi.hssf.usermodel.HSSFRichTextString;
import org.apache.poi.hssf.usermodel.HSSFWorkbook;
import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.CellStyle;
import org.apache.poi.ss.usermodel.Font;
import org.apache.poi.ss.usermodel.Row;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.Workbook;
import org.json.simple.JSONArray;
import org.json.simple.JSONObject;
import org.json.simple.parser.JSONParser;

import com.skylabs.baselogic.Solver;
import com.skylabs.baselogic.Util;

//Сервлет для экспорта данных в Excel
@WebServlet("/ExportController")
public class ExportController extends HttpServlet {
    private static final long serialVersionUID = 1L;
```

```

//Вспомогательный класс-обертка
class JustForWork {

    HttpServletRequest request;

    public JustForWork(HttpServletRequest request) {
        this.request = request;
    }

    Object get(String param) {
        return request.getParameter(param);
    }
}

//Процесс обработки запроса
@Override
protected void doGet(HttpServletRequest request,
HttpServletResponse response)
throws ServletException, IOException {

    response.setContentType("text/plain"); //Устанавливаем
MIME-тип
    response.setHeader("Content-disposition", "attachment;
filename=result.xls"); //Устанавливаем тип контента как вложение
с названием файла

    try {
        JustForWork json = new JustForWork(request);
        JSONParser parser;
        String region_name = "", product_name = "";

        //Base logic
        double perc, c, nt, pt, lt, e = 0, k = 0;

        //Получаем данные последовательно
        perc =
Double.parseDouble(String.valueOf(json.get("ndfl")));
}

```

```

        pt =
Double.parseDouble(String.valueOf(json.get("prize")));

        c =
Long.parseLong(String.valueOf(json.get("production")));

        nt = Long.parseLong(String.valueOf(json.get("count")));

        lt =
Long.parseLong(String.valueOf(json.get("normal")));

        long indexProduct =
Long.parseLong(String.valueOf(json.get("val")));

        long indexCoeff =
Long.parseLong(String.valueOf(json.get("location")));

        boolean useMrot =
Boolean.parseBoolean(String.valueOf(json.get("mrot")));
    }

    try {
        parser = new JSONParser();

        JSONObject root = (JSONObject)
parser.parse(Util.GetJson("data.json"));

        JSONArray users = (JSONArray)
root.get("entries");

        e =
Double.parseDouble(((JSONObject)users.get((int)indexProduct)).ge
t("price").toString());

        product_name =
(String)((JSONObject)users.get((int)indexProduct)).get("name");
    }

    catch(Exception ex) {
        ex.printStackTrace();
    }

    try {
        parser = new JSONParser();

        JSONObject root = (JSONObject)
parser.parse(Util.GetJson("data_regions.json"));

        JSONArray users = (JSONArray)
root.get("entries");
    }
}

```

```

        k =
Double.parseDouble(((JSONObject)users.get((int)indexCoeff)).get(
"price").toString());

        region_name =
(String)((JSONObject)users.get((int)indexCoeff)).get("name");

    }

    catch(Exception ex) {
        ex.printStackTrace();
    }

double result = Solver.Solve(perc, c, nt, pt, lt, e, k,
useMrot);

//Обработка для Excel файла
Workbook book = new HSSFWorkbook(); //Создание workbook
Sheet sheet = book.createSheet(); //Создание
страницы

//Создаем строки и заполняем их колонки данными
Row rowInfo = sheet.createRow(0);
Row rowFio = sheet.createRow(1);
Row rowPost = sheet.createRow(2);
Row rowCount = sheet.createRow(3);
Row rowPrize = sheet.createRow(4);
Row rowNdfl = sheet.createRow(8);
Row rowNorma = sheet.createRow(9);
Row rowProduct = sheet.createRow(5);
Row rowCoeff = sheet.createRow(7);
Row rowType = sheet.createRow(6);
Row rowMrot = sheet.createRow(11);
Row rowSalary = sheet.createRow(10);

Cell cell0, cell1;

CellStyle cellStyle = book.createCellStyle();

```

```

        Font font = book.createFont();
        font.setBold(true);
        cellStyle.setFont(font);

        cell0 = rowInfo.createCell(0);
        cell1 = rowInfo.createCell(1);
        cell0.setCellValue(new
HSSFRichTextString("Параметр"));
        cell1.setCellValue(new
HSSFRichTextString("Данные"));
        cell0.setCellStyle(cellStyle);
        cell1.setCellStyle(cellStyle);

        cell0 = rowFio.createCell(0);
        cell1 = rowFio.createCell(1);
        cell0.setCellValue(new HSSFRichTextString("ФИО
работника"));
        cell1.setCellValue(new
HSSFRichTextString((String)json.get("fio")));

        cell0 = rowPost.createCell(0);
        cell1 = rowPost.createCell(1);
        cell0.setCellValue(new
HSSFRichTextString("Должность работника"));
        cell1.setCellValue(new
HSSFRichTextString((String)json.get("state")));

        cell0 = rowCount.createCell(0);
        cell1 = rowCount.createCell(1);
        cell0.setCellValue(new
HSSFRichTextString("Изготовленная продукция (ед)"));
        cell1.setCellValue(new
HSSFRichTextString((String)json.get("production")));

        cell0 = rowPrize.createCell(0);
        cell1 = rowPrize.createCell(1);

```

```

        cell0.setCellValue(new HSSFRichTextString("Премия
(руб)"));

        cell1.setCellValue(new
HSSFRichTextString((String)json.get("prize")));

        cell0 = rowNdfl.createCell(0);
        cell1 = rowNdfl.createCell(1);
        cell0.setCellValue(new HSSFRichTextString("НДФЛ
%"));

        cell1.setCellValue(new
HSSFRichTextString((String)json.get("ndfl")));

        cell0 = rowNorma.createCell(0);
        cell1 = rowNorma.createCell(1);
        cell0.setCellValue(new HSSFRichTextString("Норма
(ед.)"));

        cell1.setCellValue(new
HSSFRichTextString((String)json.get("normal")));

        cell0 = rowProduct.createCell(0);
        cell1 = rowProduct.createCell(1);
        cell0.setCellValue(new HSSFRichTextString("Единица
товара для премии (ед.)"));

        cell1.setCellValue(new
HSSFRichTextString((String)json.get("count")));

        cell0 = rowCoeff.createCell(0);
        cell1 = rowCoeff.createCell(1);
        cell0.setCellValue(new
HSSFRichTextString("Районный коэффициент"));

        cell1.setCellValue(new
HSSFRichTextString(region_name + " - " + k));

        cell0 = rowType.createCell(0);
        cell1 = rowType.createCell(1);
        cell0.setCellValue(new
HSSFRichTextString("Расценка за единицу продукции (руб)"));

```

```

        cell1.setCellValue(new
HSSFRichTextString(product_name+" - " + e));

        cell0 = rowMrot.createCell(0);
        cell1 = rowMrot.createCell(1);
        cell0.setCellValue(new
HSSFRichTextString("Учитывался МРОТ?"));
        cell1.setCellValue((String) json.get("mrot") ==
"true" ? "Да" : "Нет");

        cell0 = rowSalary.createCell(0);
        cell1 = rowSalary.createCell(1);
        cell0.setCellValue(new
HSSFRichTextString("Зарплата"));

        DecimalFormat df = new DecimalFormat("#.##");

        cell1.setCellValue(df.format(result)+" руб");

        sheet.autoSizeColumn(0);
        sheet.autoSizeColumn(1);

        OutputStream out = response.getOutputStream();
//Возвращаем поток данных запроса
        book.write(out); //Записываем наши данные о файле
в выходной поток
        book.close();

    }

    catch(Exception ex) {
        ex.printStackTrace();
    }

}

}

```

## **UTILHELPER.JAVA**

```
package com.skylabs.baselogic;

public interface UtilHelper {

    //Готовые строки для файлов JSON, если их нет в ФС

    static final String DEFAULT_REGIONS =
"{\"entries\": [{\"name\":\"Республика
Башкортостан\", \"price\":1.15}, {\"name\":\"Ростовская
область\", \"price\":1.1}, {\"name\":\"Республика
Дагестан\", \"price\":1.12}, {\"name\":\"Республика
Калмыкия\", \"price\":1.3}, {\"name\":\"Ставропольский
край\", \"price\":1.15}]}";

    static final String DEFAULT_PRODUCTS =
"{\"entries\": [{\"name\":\"Компрессор\", \"price\":1000}, {\"name\"
:\"Вентилятор\", \"price\":300}, {\"name\":\"Турбина\", \"price\":
\"400\"}, {\"name\":\"Сопло\", \"price\":100}, {\"name\":\"Смесител
ь\", \"price\":200}]}";

    static final String DEFAULT_USERS =
"{\"users\": [{\"password\":\"admin\", \"admin\":true, \"userna
me\":\"admin\"}]}";

}
```

## *REGISTERCONTROLLER.JAVA*

```
package com.skylabs.controllers;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.json.simple.JSONArray;
import org.json.simple.JSONObject;
import org.json.simple.parser.JSONParser;
import org.json.simple.parser.ParseException;

import com.skylabs.baselogic.Util;

//Сервлет для регистрации
@WebServlet("/RegisterController")
public class RegisterController extends HttpServlet {

    private static final long serialVersionUID = 1L;

    public RegisterController() {
        super();
    }

    //Процесс обработки запроса на регистрацию нового
    //пользователя

    protected void doGet(HttpServletRequest request,
                         HttpServletResponse response) throws ServletException,
                         IOException {
        request.setCharacterEncoding("UTF-8");
```

```

        response.setCharacterEncoding("UTF-8");
        response.setContentType("text/html");

        String username = request.getParameter("username");
        //Получаем параметры из запроса

        String password = request.getParameter("password");

        PrintWriter writer = response.getWriter();

        if (Register(username, password)) { //Пытаемся
зарегистрировать

            try {
                writer.println("Аккаунт с ником "+username+
успешно создан!");
            } finally {
                writer.close();
            }
        }

        else { //Если пользователь уже такой есть или что-то
пошло не так

            try {
                writer.println("Не удается создать аккаунт с
логином "+username+"!");
            } finally {
                writer.close();
            }
        }
    }

    //Регистрация нового пользователя. True - успех, False - нет
@SuppressWarnings("unchecked")
boolean Register(String username, String password) {

    String json = Util.GetJson("users.json"); //Получаем
список пользователей

    try {

```

```

JSONParser parser = new JSONParser();
JSONObject root = (JSONObject) parser.parse(json);
JSONArray users = (JSONArray) root.get("users");

for(Object entry : users) {
    JSONObject user = (JSONObject) entry;
    if (user.get("username").equals(username))
return false;
}

JSONObject new_user = new JSONObject(); //Создаем
новый узел JSON для пользователя и записываем туда данные
new_user.put("username", username);
new_user.put("password", password);
new_user.put("admin", "false");

users.add(new_user); //Добавляем нового
пользователя в узел пользователей JSON

JSONObject new_root = new JSONObject();
new_root.put("users", users);

Util.SaveJson(new_root, "users.json"); //Сохраняем
обновленный список пользователей в файл

} catch (ParseException e) {
    e.printStackTrace();
    return false;
}
return true;
}

```

## AUTH.CSS

```
* {
    box-sizing: border-box;
}

body {
    background-color: whitesmoke;
}

.content {
    width: 300px;
    height: 200px;
    /*box-shadow: 0 0 10px darkgray;*/
    background: linear-gradient(to top left, #6742A0, #A53E5E);
    margin-top: 250px;
    margin-left: 300px;
    border-radius: 12px;
    text-align: center;
    padding: 10px;
    transform: scale(0);
    opacity: .9;
}
.animation {
    transition: .5s all ease-in-out;
    transform: scale(1);
}

.content h2 {
    margin: 0;
}

.buttons {
    display: flex;
    flex-direction: column;
    align-items: center;
}

.btn {
    margin-top: 20px;
    width: 200px;
    border-radius: 12px;
    font-weight: bold;
}
.btn-present {
    background: linear-gradient(to top right, #3D2255, #5C1E54);
    outline: none !important;
    border: none;
    transition: all ease-in-out .25s;
}
.btn-present:focus {
    outline: none;
}
```

```
.btn-present:hover {  
    transform: scale(1.1);  
}  
  
.vanta-background {  
    position: absolute;  
    height: 100%;  
    width: 100%;  
    left: 0;  
    top: 0;  
}  
  
h2 {  
    color: white;  
}
```

## *STYLE.CSS*

```
.mr0t-check {  
    margin: 50px 0 50px 0;  
    font-size: 18px;  
}  
.root {  
    padding-top: 50px;  
}  
  
.container footer {  
    text-align: center;  
    margin-top: 138px;  
}  
  
.custom-select {  
    height: 38px;  
}  
  
.register_form, .login_form {  
    box-shadow: 0 0 10px 10px;  
    width: 400px;  
    height: 300px;  
}  
  
.btn-user {  
    outline-color: green;  
}  
  
.adminEntry {  
    display: flex;  
    flex-direction: row;  
    margin-bottom: 10px;  
}  
  
.input-price {  
    width: 100px;  
    margin: 0 10px 0 10px;  
}
```

## ADMIN.JS

```
/*
Этот JS файл необходим для корректной работы админского
функционала по редактированию
коэффициентов продукции и регионов. Здесь имеется несколько
функций-обработчиков для событий
кнопок. Каждый из них выполняет свою задачу для обеспечения
возможности редактирования.
*/



//Переменные для работы
var adminUrl = "/EditController"
var admin_entries = []
var default_name = ""
var default_price = 0
var edit_type = ""


//Линковка админ-событий
function adminEvents() {
    document.getElementById("actionToEditProductions").addEventListener("click", (event) => adminEdit(event, 0));
    document.getElementById("actionToEditRegions").addEventListener("click", (event) => adminEdit(event, 1));
    document.getElementById("adminAddNew").addEventListener("click", adminAddNewElement);
    document.getElementById("adminSave").addEventListener("click", adminSave);
}

//Добавление нового элемента
function adminAddNewElement() {
    admin_entries.push({name: default_name, price: default_price});
    adminRender();
}

//Пересобирание DOM-элемента
function adminRender() {
    $("#adminContent").empty();
    var i = 0;
    admin_entries.forEach((entry) => {
        var new_entry = document.createElement("div");
        var input_name = document.createElement("input");
        var input_price = document.createElement("input");
        var remove_button = document.createElement("button");
        new_entry.className = "adminEntry form-group";
        input_name.className = "form-control";
        input_name.value = entry.name;
        input_name.type = "text";
        input_name.onchange = adminNameChange
        input_name.id = "iname"+i
        input_price.className = "form-control input-price";
        new_entry.appendChild(input_name);
        new_entry.appendChild(input_price);
        new_entry.appendChild(remove_button);
        document.getElementById("adminContent").appendChild(new_entry);
    });
}
```

```

        input_price.value = entry.price;
        input_price.type = "text";
        input_price.onchange = adminPriceChange;
        input_price.id = "iprice"+i;
        remove_button.className = "btn btn-danger";
        remove_button.innerText = "Удалить";
        remove_button.type = "button";
        remove_button.id = "rbutton"+i;
        remove_button.addEventListener("click", (event) =>
adminRemove(event));
        new_entry.appendChild(input_name);
        new_entry.appendChild(input_price);
        new_entry.appendChild(remove_button);
        $("#adminContent").append(new_entry);
        i++;
    });
}

//Смена название записи
function adminNameChange(event) {
    var str = String(event.srcElement.id);
    var id = parseInt(str.substr(5,str.length - 5));
    admin_entries[id].name = event.srcElement.value;
}

//Изменение значения цены
function adminPriceChange(event) {
    var str = String(event.srcElement.id);
    var id = parseInt(str.substr(6,str.length - 6));
    admin_entries[id].price = event.srcElement.value;
}

//Удаление записи
function adminRemove(event) {
    var str = String(event.srcElement.id);
    var id = parseInt(str.substr(7,str.length - 7));
    admin_entries.splice(id, 1);
    adminRender();
}

//Сохранение результатов работы и отправка на сервер
function adminSave() {

    varisOk = true

    var root = {
        "entries": [
            ]
    }
    admin_entries.forEach((entry) => {
        if (!entry.name || isNaN(parseFloat(entry.price))) {
            isOk = false;
    }
}

```

```

        }
        root.entries.push({ "name": entry.name, "price": entry.price });
    });

    if (isOk) {

        $.post(adminUrl+"?type="+edit_type+"&secret_key="+secret_key+"&login"+login, JSON.stringify(root), function(response) {

            showHelp(null, "Информация", "Запрос обработан!");
        });

    } else {
        showHelp(null, "Ошибка", "Пожалуйста, заполните все поля корректными данными.");
    }
}

//Начало редактирования коэффициентов
function adminEdit(event, type) {
    var get_params = type == 0 ? "?type=products" : "?type=regions";
    if (type == 0) {
        default_name = "Новый";
        default_price = 100;
        edit_type = "products";
    } else {
        default_name = "Республика X";
        default_price = 1;
        edit_type = "regions";
    }
    $.get(adminUrl+get_params, function(response) {
        admin_entries = []
        var root = JSON.parse(response);
        root.entries.forEach(entry => {
            admin_entries.push(entry);
        });
        adminRender();
        $("#modalAdmin").modal("toggle");
    });
}

//Вызываем линкову событий с обработчиками
adminEvents();

```

## AUTH.JS

```
/*
Данный JS файл необходим для обеспечения возможности авторизации и регистрации пользователя.
Здесь имеется 2 функция-обработчика событий - для регистрации и авторизации. В обоих используется
AJAX для обеспечения лучшей интерактивности с пользователем.
*/



//URL адреса сервлетов сервера
var login_url = "/AuthController";
var register_url = "/RegisterController";

//Линковка кнопки регистрации для вызова модального окна регистрации
$("#btnRegister").click(function(event) {
    $("#modalRegister").modal("toggle");
});

//Линковка кнопки авторизации для вызова модального окна авторизации
$("#btnAuth").click(function(event) {
    $("#modalAuth").modal("toggle");
});

//При нажатии на кнопку регистрации
$("#btnRegisterAction").click(function(event) {
    event.preventDefault(); //Отменяем дефолтное событие формы
    if ($("#passwordInput").val() ==
        $("#passwordInputRepeat").val()) //Проверяем, чтобы пароли совпадали (пароль = повторить пароль)
    {
        var get_parameters =
"?username="+$("#registerInput").val()+"&password="+$("#passwordInput").val();
        $.get(register_url+get_parameters, function(response) {
//Делаем GET AJAX запрос на сервер
            showInfo(response);
            if (response.substring(0, 7) == "Аккаунт")
                $("#modalRegister").modal("toggle");
        });
    }
    else {
        showInfo("Пароли не совпадают!");
    }
});

//Обработка авторизации
$("#btnAuthAction").click(function(event) {
    event.preventDefault();
```

```

var get_parameters =
"?username="+$("#loginInput").val()+"&password="+$("#PasswordLog
inInput").val();
$.get(login_url+get_parameters, function(response) {
    if (response == 1) { //Если авторизовались
        window.location = "engine.jsp"; //Переходим на
страницу работы с калькулятором
    } else showInfo(response); //Иначе показываем модальное
окно с информацией об ошибке
});
});

//Функция для работы с модальным окном для вывода определенной
служебной информации пользователю
function showInfo(info) {
    $("#modalShowText").text(info);
    $("#modalShow").modal("toggle");
}

//Добавляем анимации
$( document ).ready(function() {
    $("#authRoot").addClass("animation");
});

```

## MAIN.JS

```
/*
Данный JS файл необходим для работы основного приложения. Здесь
происходит
основная работа по обеспечению интерактивности и получению
необходимых
результатов для пользователя.
*/



//Переменные для работы
var secret_key = "";
var login = "";


var manual = "Данный калькулятор предназначен для расчёта
сдельно-премиальной зарплаты.<br/>" +
"Сдельно-премиальная оплата труда - это форма, которая
предполагает не только исчисление прямого заработка с учетом
фактических результатов труда, но и установление надбавок
(премий) за выполнение и перевыполнение плановых
показателей.<br>" +
"Для расчёта необходимо ввести все значения.<br/>" +
"Расчёт осуществляется с учётом <strong>МРОТ, НДФЛ.</strong>";;
var developers = "Калькулятор был разработан студентами группы
<strong>ПИ-221</strong>: " + "<br/>Рафиков Даниил"
+"<br/>Катасонов Серафим" +"<br/>Гибадуллина Элина" +"<br/>Газин
Даниэль";


//URL адреса серверов сервера
var serverUrl = "/CalculatorController";
var exportUrl = "/ExportController";
var exitUrl = "/ExitController";


//Линковка событий с обработчиками
function linkEvents() {

    document.getElementById("submitButton").addEventListener("click",
, calculate);

    document.getElementById("actionNewFile").addEventListener("click",
, newFile);

    document.getElementById("actionExit").addEventListener("click",
exit);

    document.getElementById("actionToExcel").addEventListener("click",
, toExcel);

    document.getElementById("actionHelp").addEventListener("click",
(event) => showHelp(event, "Справка", manual));

    document.getElementById("actionDevelopers").addEventListener("cl
ick",
(event) => showHelp(event, "Разработчики", developers));
}
```

```

}

//Получаем данные из форм
function getData() {
    var data = {
        fio: document.getElementById("fio").value,
        state: document.getElementById("state").value,
        val: document.getElementById("val").selectedIndex,
        production:
            parseInt(document.getElementById("production").value),
        prize:
            parseFloat(document.getElementById("prize").value),
        count: parseInt(document.getElementById("count").value),
        ndfl: parseFloat(document.getElementById("ndfl").value),
        normal:
            parseInt(document.getElementById("normal").value),
        location:
            document.getElementById("location").selectedIndex,
        mrot: document.getElementById("mrot").innerText ==
        "Учитывать" ? true : false
    }

    return data;
}

//Валидация данных из форм. Чтобы не было пустых строк и
//неправильных данных для будущих вычислений на стороне сервера
function validateData(data) {
    if (!data.fio || !data.state)
        return false;
    if (isNaN(data.val) || isNaN(data.production) ||
    isNaN(data.normal) || isNaN(data.count))
        return false;
    if (isNaN(data.ndfl) || isNaN(data.prize))
        return false;
    return true;
}

//Запрос на вычисление заработной платы на основе введенных
//данных
function calculate(event) {
    event.preventDefault();

    var data = getData();

    if (validateData(data)) {

$.post(serverUrl+"?secret_key="+secret_key+"&login"+login,
JSON.stringify(data), function(response) {
    showHelp(null, "Результат вычисления", response);
});
    } else {

```

```

        showHelp(null, "Ошибка", "Пожалуйста, заполните все
поля корректными данными.")
    }
}

//Очистка формы
function newFile(event) {
    document.getElementById("fio").value = "";
    document.getElementById("state").value = "";
    document.getElementById("production").value = "";
    document.getElementById("prize").value = "";
    document.getElementById("count").value = "";
    document.getElementById("ndfl").value = "";
    document.getElementById("normal").value = "";
}

//Запрос на преобразование данных формы в Excel файл
function toExcel(event) {
    var data = getData();
    if (validateData(data)) {
        str =
"?fio="+data.fio+"&state="+data.state+"&val="+data.val+"&product
ion="+data.production+"&prize="+data.prize+"&count="+data.count+
"&ndfl="+data.ndfl+"&normal="+data.normal+"&location="+data.loca
tion+"&mrot="+data.mrot;

        downloadFile(exportUrl+str+"&secret_key="+secret_key+"&login
"+login);
    } else {
        showHelp(null, "Ошибка", "Пожалуйста, заполните все
поля корректными данными.")
    }
}

//Дополнительная функция для скачивания файлов с сервера
function downloadFile(urlToSend) {
    var req = new XMLHttpRequest();
    req.open("GET", urlToSend, true);
    req.responseType = "blob";
    req.onload = function (event) {
        var blob = req.response;
        var link=document.createElement('a');
        link.href=window.URL.createObjectURL(blob);
        link.download="result.xls";
        link.click();
    };

    req.send();
}

//Вызов служебного модального окна с информацией
function showHelp(event, arg0, arg1 = "") {
    document.getElementById("modalShowText").innerHTML = arg1
}

```

```
document.getElementById("modalShowTitle").innerHTML = arg0;
$("#modalShow").modal("toggle");
}

//Запрос на выход из сессии
function exit() {
    $.get(exitUrl, function(response) {
        if (response == 1) {
            window.location = "index.jsp";
        } else showInfo("Что-то пошло не так...");
    });
}

//Линковка событий с обработчиками
linkEvents()
```

## ENGINE.JSP

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8" import="com.skylabs.baselogic.Util"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Калькулятор</title>
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm
" crossorigin="anonymous">
    <link rel="stylesheet" href="css/style.css">
    <link rel="icon" type="image/ico" href="picture/icon.ico">
</head>
<body>
    <c:if test="${login}">
        <div id="root">
            <header>
                <nav class="navbar navbar-expand-lg navbar-light bg-
light">
                    <div class="container">
                        <a class="navbar-brand" href="#">Калькулятор</a>
                        <button class="navbar-toggler" type="button"
data-toggle="collapse" data-target="#navbarSupportedContent"
aria-controls="navbarSupportedContent" aria-expanded="false"
aria-label="Toggle navigation">
                            <span class="navbar-toggler-icon"></span>
                        </button>

                        <div class="collapse navbar-collapse"
id="navbarSupportedContent">
                            <ul class="navbar-nav mr-auto">
                                <li class="nav-item dropdown">
                                    <a class="nav-link dropdown-toggle"
href="#" id="navbarDropdown" role="button" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">
                                        Файл
                                    </a>
                                    <div class="dropdown-menu" aria-
labelledby="navbarDropdown">
                                        <a class="dropdown-item" href="#">
                                            Новый</a>
                                            <div class="dropdown-divider"></div>
                                            <a class="dropdown-item" href="#">
                                                Выйти</a>
                                        </div>

```

```

        </li>
        <li class="nav-item dropdown">
            <a class="nav-link dropdown-toggle"
href="#" id="navbarDropdown" role="button" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">
                Экспорт
            </a>
            <div class="dropdown-menu" aria-
labelledby="navbarDropdown">
                <a class="dropdown-item" href="#" id="actionToExcel">В Excel</a>
                </div>
            </li>
            <li class="nav-item dropdown">
                <a class="nav-link dropdown-toggle"
href="#" id="navbarDropdown" role="button" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">
                    О приложении
                </a>
                <div class="dropdown-menu" aria-
labelledby="navbarDropdown">
                    <a class="dropdown-item" href="#" id="actionHelp">Справка</a>
                    <!-- <a class="dropdown-item"
href="#" id="actionAbout">О калькуляторе</a> -->
                    <div class="dropdown-divider"></div>
                    <a class="dropdown-item" href="#" id="actionDevelopers">Разработчики</a>
                    </div>
                </li>
                <c:if test="${admin}">
                    <li class="nav-item dropdown">
                        <a class="nav-link dropdown-toggle"
href="#" id="navbarDropdown" role="button" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">
                            Редактировать
                        </a>
                        <div class="dropdown-menu" aria-
labelledby="navbarDropdown">
                            <a class="dropdown-item"
href="#" id="actionUpEditProductions">Продукция</a>
                            <div class="dropdown-
divider"></div>
                            <a class="dropdown-item"
href="#" id="actionUpEditRegions">Коэффициенты</a>
                            </div>
                        </li>
                    </c:if>
                </ul>
                <form class="form-inline my-2 my-lg-0">
                    <c:if test="${admin}">

```

```

        <button class="btn btn-outline-danger
my-2 my-sm-0" type="submit" disabled>Администратор <%
session.getAttribute("username") %></button>
    </c:if>
    <c:if test="${!admin}">
        <button class="btn btn-outline-success
my-2 my-sm-0" type="submit" disabled>Пользователь <%
session.getAttribute("username") %></button>
    </c:if>
</form>
</div>
</div>
</nav>
</header>
<div class="container root">
<form>
    <div class="row">
        <div class="col">
            <div class="form-group">
                <label for="fio">ФИО работника:</label>
                <input required class="form-control"
type="text" id="fio" placeholder="Иванов Петр Князевич">
            </div>
            <div class="form-group">
                <label for="state">Должность
работника:</label>
                <input required class="form-control"
type="text" id="state" placeholder="Слесарь">
            </div>
            <div class="form-group">
                <label for="val">Расценка за единицу
продукции:</label>
                <select id="val" class="custom-select
custom-select-sm">
                    <%
                        for(String entry :
Util.GetProducts()) {
                            out.println("<option>" + entry + "</option>");
                        }
                    <%
                </select>
            </div>
            <div class="form-group">
                <label for="production">Изготовленная
продукция (ед.):</label>
                <input required class="form-control"
type="text" id="production" placeholder="0">
            </div>
            <div class="form-group">
                <label for="prize">Премия
(руб.):</label>

```

```

                <input required class="form-control"
type="text" id="prize" placeholder="1000">
            </div>
        </div>
        <div class="col">
            <div class="form-group">
                <label for="count">Единица товара для
премии (ед.):</label>
                <input required class="form-control"
type="text" id="count" placeholder="5">
            </div>
            <div class="form-group">
                <label for="ndfl">НФДЛ (%) :</label>
                <input required class="form-control"
type="text" id="ndfl" placeholder="13">
            </div>
            <div class="form-group">
                <label for="normal">Норма (ед.):</label>
                <input required class="form-control"
type="text" id="normal" placeholder="1">
            </div>
            <div class="form-group">
                <label for="location">Районный
коэффициент:</label>
                <select id="location" class="custom-
select custom-select-sm">
                    <%
                        for(String entry :
Util.GetLocations()) {
                            out.println("<option>" + entry + "</option>");
                    }
                    %>
                </select>
            </div>
            <div class="form-group">
                <label for="mrot">Учет МРОТа</label>
                <select id="mrot" class="custom-select
custom-select-sm">
                    <option selected>Не
учитывать</option>
                    <option selected>Учитывать</option>
                </select>
            </div>
            <button type="submit" class="btn-lg btn-
primary" style="float:right; margin-top: 10px;" id="submitButton">Рассчитать</button>
        </div>
    </div>
</form>
</div>

<div class="container">

```

```

<footer>
    УГАТУ 2021 - Курсовая работа - Вариант 1
</footer>
</div>

<!-- Modals -->
<div class="modal fade" id="modalShow" tabindex="-1"
role="dialog" aria-labelledby="exampleModalCenterTitle" aria-
hidden="true">
    <div class="modal-dialog modal-dialog-centered"
role="document">
        <div class="modal-content">
            <div class="modal-header">
                <h5 class="modal-title"
id="modalShowTitle">Справка</h5>
                <button type="button" class="close" data-
dismiss="modal" aria-label="Close">
                    <span aria-hidden="true">&times;</span>
                </button>
            </div>
            <div class="modal-body">
                <p id="modalShowText">
                    Lorem ipsum dolor sit amet consectetur,
                    adipisicing elit. Aut quod quo obcaecati preferendis soluta
                    amet, deleniti earum velit culpa numquam laudantium ullam
                    mollitia in, iusto repellendus commodi ipsum pariatur quaerat?
                </p>
            </div>
            <div class="modal-footer">
                <button type="button" class="btn btn-secondary"
data-dismiss="modal">Закрыть</button>
                <!-- <button type="button" class="btn btn-
primary">Save changes</button> -->
            </div>
        </div>
    </div>
</div>
</c:if>
<c:if test="${!login}">
    <h2>Доступ запрещён!</h2>
</c:if>
<c:if test="${admin}">
    <div class="modal fade" id="modalAdmin" tabindex="-1"
role="dialog" aria-labelledby="exampleModalCenterTitle" aria-
hidden="true">
        <div class="modal-dialog modal-dialog-centered"
role="document">
            <div class="modal-content">
                <div class="modal-header">
                    <h5 class="modal-title"
id="modalShowTitle">Редактирование</h5>
                    <button type="button" class="close" data-
dismiss="modal" aria-label="Close">

```

```

        <span aria-hidden="true">&times;</span>
    </button>
</div>
<div class="modal-body" id="adminContent">
    <div class="adminEntry form-group">
        <input required class="form-control" type="text" value="Компрессор" id="val_name_0">
            <input required class="form-control input-price" type="text" value="100" id="val_price_0">
                <button type="button" class="btn btn-danger">Удалить</button>
            </div>
        </div>
        <div class="modal-footer">
            <button type="button" class="btn btn-primary" id="adminAddNew">Добавить новый</button>
            <button type="button" class="btn btn-secondary" id="adminSave" data-dismiss="modal">Закрыть и сохранить</button>
            <!-- <button type="button" class="btn btn-primary">Save changes</button> -->
        </div>
    </div>
</div>
</c:if>

<script src="https://code.jquery.com/jquery-3.6.0.min.js" integrity="sha256-/xUj+3OJU5yExlq6GSYGSk7tPXikynS7ogEvDej/m4=" crossorigin="anonymous"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js" integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q" crossorigin="anonymous"></script>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js" integrity="sha384-JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQQxSffWpi1MquVdAyjUar5+76PVCmYl" crossorigin="anonymous"></script>
<script src="js/main.js"></script>
<c:if test="${admin}">
    <script src="js/admin.js"></script>
</c:if>
</div>
</body>
</html>

```

## INDEX.JSP

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Авторизация</title>
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm
" crossorigin="anonymous">
    <link rel="stylesheet" href="css/auth.css">
    <link rel="icon" type="image/ico" href="picture/icon.ico">
</head>
<body>
    <div class="vanta-background">

        </div>

        <div class="container content" id="authRoot">
            <h2>Калькулятор</h2>
            <div class="buttons">
                <button class="btn btn-primary btn-present"
id="btnRegister">Регистрация</button>
                <button class="btn btn-primary btn-present"
id="btnAuth">Авторизация</button>
            </div>
        </div>

        <!-- Modals -->
        <div class="modal fade" id="modalAuth" tabindex="-1"
role="dialog" aria-labelledby="exampleModalCenterTitle" aria-
hidden="true">
            <div class="modal-dialog modal-dialog-centered"
role="document">
                <div class="modal-content">
                    <div class="modal-header">
                        <h5 class="modal-title"
id="modalShowTitle">Авторизация</h5>
                        <button type="button" class="close" data-
dismiss="modal" aria-label="Close">
                            <span aria-hidden="true">&times;</span>
                        </button>
                    </div>
                    <div class="modal-body">
                        <form action="#" method="POST">
```

```

        <div class="mb-3">
            <label for="emailInput" class="form-label">Ваш Логин:</label>
                <input required name="username" type="text" class="form-control" id="loginInput" aria-describedby="emailHelp">
            </div>
            <div class="mb-3">
                <label for="PasswordLoginInput" class="form-label">Пароль:</label>
                    <input required name="password" type="password" class="form-control" id="PasswordLoginInput">
                </div>
                <div class="mb-3 form-check">
                    <input name="password" type="checkbox" class="form-check-input" id="exampleCheck1">
                    <label class="form-check-label" for="exampleCheck1">Запомнить</label>
                </div>
                <button type="submit" class="btn btn-primary" id="btnAuthAction">Авторизоваться</button>
            </form>
        </div>
        <div class="modal-footer">
            <button type="button" class="btn btn-secondary" data-dismiss="modal">Закрыть</button>
            <!-- <button type="button" class="btn btn-primary">Save changes</button> -->
        </div>
    </div>
</div>

<div class="modal fade" id="modalRegister" tabindex="-1" role="dialog" aria-labelledby="exampleModalCenterTitle" aria-hidden="true">
    <div class="modal-dialog modal-dialog-centered" role="document">
        <div class="modal-content">
            <div class="modal-header">
                <h5 class="modal-title" id="modalRegister">Регистрация</h5>
                <button type="button" class="close" data-dismiss="modal" aria-label="Close">
                    <span aria-hidden="true">&times;</span>
                </button>
            </div>
            <div class="modal-body">
                <form action="#" method="POST">
                    <div class="mb-3">
                        <label for="emailInput" class="form-label">Ваш Логин:</label>

```

```

        <input required name="username"
type="text" class="form-control" id="registerInput" aria-
describedby="emailHelp">
    </div>
    <div class="mb-3">
        <label for="passwordInput" class="form-
label">Пароль:</label>
            <input required name="password"
type="password" class="form-control" id="passwordInput">
        </div>
        <div class="mb-3">
            <label for="passwordInputRepeat"
class="form-label">Повторить пароль:</label>
            <input required name="password1"
type="password" class="form-control" id="passwordInputRepeat">
        </div>
            <button type="submit" class="btn btn-
primary" id="btnRegisterAction">Создать аккаунт</button>
        </form>
    </div>
    <div class="modal-footer">
        <button type="button" class="btn btn-secondary"
data-dismiss="modal">Закрыть</button>
        <!-- <button type="button" class="btn btn-
primary">Save changes</button> -->
    </div>
</div>
</div>

<!-- Modals -->
<div class="modal fade" id="modalShow" tabindex="-1"
role="dialog" aria-labelledby="exampleModalCenterTitle" aria-
hidden="true">
    <div class="modal-dialog modal-dialog-centered"
role="document">
        <div class="modal-content">
            <div class="modal-header">
                <h5 class="modal-title"
id="modalShowTitle">Информация</h5>
                <button type="button" class="close" data-
dismiss="modal" aria-label="Close">
                    <span aria-hidden="true">&times;</span>
                </button>
            </div>
            <div class="modal-body">
                <p id="modalShowText"></p>
            </div>
            <div class="modal-footer">
                <button type="button" class="btn btn-secondary"
data-dismiss="modal">Закрыть</button>
            </div>
        </div>
    </div>

```

```

        </div>
    </div>

    <script src="https://code.jquery.com/jquery-3.6.0.min.js"
integrity="sha256-/xUj+3OJU5yExlq6GSYGSKh7tPXikynS7ogEvDej/m4="
crossorigin="anonymous"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd
/popper.min.js" integrity="sha384-
ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q
" crossorigin="anonymous"></script>
    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap
.min.js" integrity="sha384-
JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQxSffWpi1MquVdAyjUar5+76PVCmYl
" crossorigin="anonymous"></script>
    <script src="js/auth.js"></script>

    <script
src="https://cdnjs.cloudflare.com/ajax/libs/three.js/r121/three.
min.js"></script>
    <script
src="https://cdn.jsdelivr.net/npm/vanta@latest/dist/vanta.globe.
min.js"></script>
    <script>
        VANTA.GLOBE({
            el: ".vanta-background",
            mouseControls: true,
            touchControls: true,
            gyroControls: false,
            minHeight: 200.00,
            minWidth: 200.00,
            scale: 1.00,
            scaleMobile: 1.00
        })
    </script>
</body>
</html>

```

## **ПРИЛОЖЕНИЕ 3**

### *ПРОГРАММНЫЙ КОД ТЕСТ*

#### *REGISTERTEST*

```
package com.skylabs.controllers;

import static org.junit.Assert.*;
import org.junit.BeforeClass;
import org.junit.Test;

public class RegisterTest {
    static RegisterController testRegister;
    @BeforeClass
    public static void setUpBeforeClass() throws Exception {
        testRegister= new RegisterController ();
    }

    @Test
    public void test() {
        assertFalse(testRegister.Register("admin", "admin"));
    }
}
```

## **ПРИЛОЖЕНИЕ 4**

### *ПРОГРАММНЫЙ КОД ТЕСТ*

#### *SOLVERTEST*

```
package com.skylabs.baselogic;

import static org.junit.Assert.*;
import org.junit.Test;

public class SolverTest {

    @Test
    public void test() {
        Double result = Solver.Solve(13, 15, 5, 1000, 10, 1000,
1.15f, true);
        assertTrue(result == 16007.999668121338);
    }

}
```

## **ПРИЛОЖЕНИЕ 5**

### *ПРОГРАММНЫЙ КОД ТЕСТ*

#### *AUTHTEST*

```
package com.skylabs.controllers;

import static org.junit.Assert.*;
import org.junit.BeforeClass;
import org.junit.Test;

public class AuthTest {

    static AuthController AC;
    @BeforeClass
    public static void setUpBeforeClass() throws Exception {
        AC = new AuthController();
    }

    @Test
    public void test() {
        assertTrue(AC.Authorize("admin", "admin") == 2);
    }
}
```

## ПРИЛОЖЕНИЕ 6

### *ПРОГРАММНЫЙ КОД ТЕСТ*

#### *SAVEJSONTEST*

```
package com.skylabs.baselogic;

import static org.junit.Assert.*;
import java.io.File;
import java.io.IOException;
import org.junit.Before;
import org.junit.BeforeClass;
import org.junit.Test;

public class SaveJsonTest {

    String fileName = "test.json";

    @Test
    public void test() {
        try {
            Util.SaveString(fileName, "test");
            File fl = new File(fileName);
            assertEquals(fl.exists(), true);
            fl.delete();
        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
            fail("Ошибка. Файл не создан");
        }
    }
}
```

## **ПРИЛОЖЕНИЕ 7**

### **ФГБОУ ВО УФИМСКИЙ ГОСУДАРСТВЕННЫЙ АВИАЦИОННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

СОГЛАСОВАНО  
Доцент кафедры АСУ ФГБОУ  
ВО УГАТУ  
Личная расшифровка  
подпись подписи  
17.05.2021

УТВЕРЖДАЮ  
Студент группы ПИ-221  
ФИРТ ФГБОУ УГАТУ, модератор  
Личная расшифровка  
подпись подписи  
17.05.2021

### **Калькулятор сдельно-премиальной зарплаты**

#### **Руководство оператора**

#### **ЛИСТ УТВЕРЖДЕНИЯ**

**1304.300001.000 34-ЛУ**

СОГЛАСОВАНО

Представитель  
команды разработчиков

Доцент кафедры АСУ  
ФГБОУ ВО УГАТУ  
Личная расшифровка  
подпись подписи  
17.05.2021

Студент группы ПИ-221  
ФИРТ ФГБОУ ВО УГАТУ, модератор  
Личная расшифровка  
подпись подписи  
17.05.2021

Изм. N							
Полн. И.дата	Полн. И.дата	Взам. И.дата					

2021

Утвержден  
1304.300001.000 34-ЛУ

ФГБОУ ВО УФИМСКИЙ ГОСУДАРСТВЕННЫЙ АВИАЦИОННЫЙ  
ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

**Программный продукт  
«Калькулятор сдельно-премиальной зарплаты»  
Руководство оператора**

**1304.300001.000 34**

**Листов 15**

Нбр.№ помл.	Помл. №	В здам. кол-во.№	Мбр.№ куб.№	Помл. № куб.№

2021

## **АННОТАЦИЯ**

Данный документ является руководством пользователя для программного продукта «Калькулятор сдельно-премиальной зарплаты». Документ разработан в рамках курсового проекта в соответствии с ГОСТ 19.505-79.

**СОДЕРЖАНИЕ**

1. Назначение программы.....	5
2. Условия выполнения программы.....	6
3. Выполнение программы.....	7
4. Сообщения оператору.....	12

## **1. Назначение программы**

Для программы определены следующие роли пользователей: простой пользователь и администратор.

Функциональное назначение программного продукта: программа предназначена для автоматизации расчета заработной платы для сотрудника-пользователя и возможность изменения настроек для администратора.

Эксплуатационное назначение: сотрудники, работающие в структурном подразделении «Цех», например, рабочий.

Данный калькулятор предназначен для предприятия ООО «Авиа-строй».

## 2. Условия выполнения программы

Для работы с Калькулятором не требуется установка дополнительного программного обеспечения.

Пользователь работает с программой через веб-браузер, где у него есть доступ к веб-приложению, расположенному по определённому url-адресу: <https://pi221team1-course.herokuapp.com/>.

Для работы с Калькулятором у пользователя должна быть любая из следующих операционных систем и установлен браузер:

- Windows 10, а именно браузеры: Edge, Google Chrome, Mozilla Firefox;
- Linux Ubuntu - браузеры Mozilla Firefox, Google Chrome;
- Linux Mint - браузеры Mozilla Firefox, Google Chrome.

### 3. Выполнение программы

Программа выполняется посредством успешного перехода по адресу <https://pi221team1-course.herokuapp.com/>. Далее пользователя встречает окно с выбором авторизации или регистрации пользователя (рис. 1).

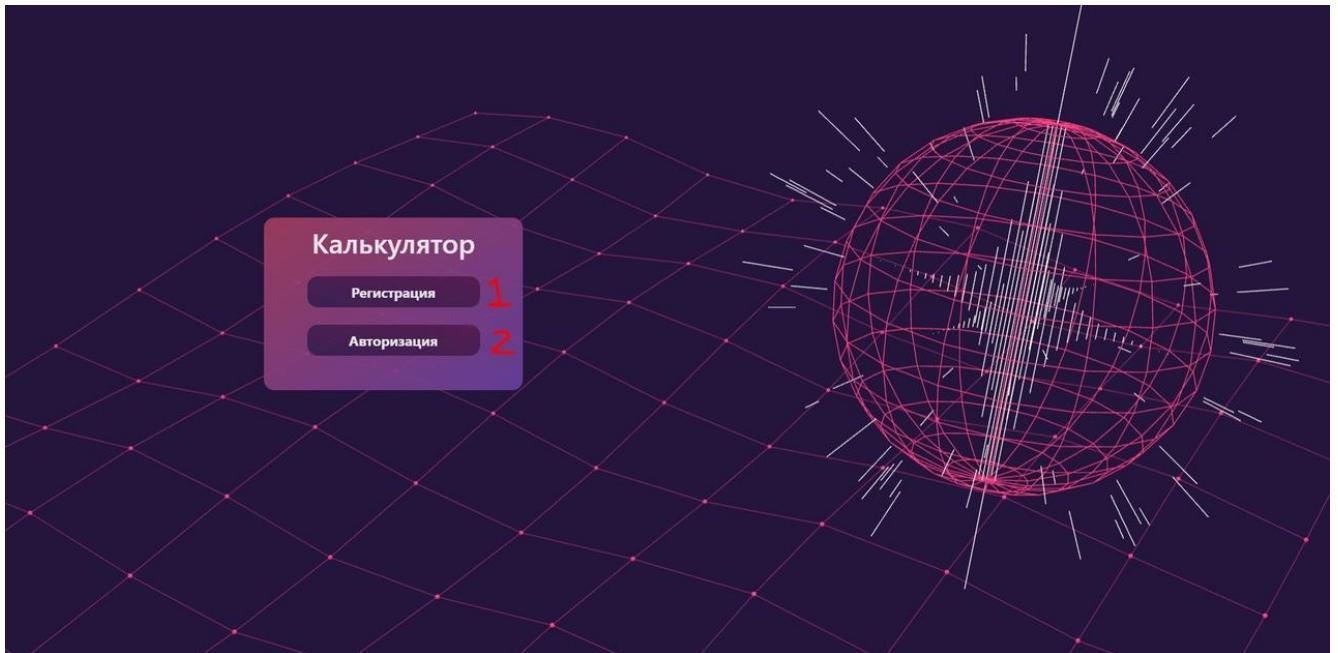


Рисунок 1 - Окно авторизации или регистрации пользователя

В случае, если у пользователя нет аккаунта в приложении, ему необходимо будет зарегистрироваться, нажав на кнопку «Регистрация» (1).

Если у пользователя есть аккаунт в приложении, он может авторизоваться, нажав на кнопку «Авторизация» (2).

После успешной регистрации/авторизации пользователя, произойдет переход на основную страницу программы (рис.2)

Калькулятор    Файл ▾    Экспорт ▾    О приложении ▾

Пользователь логин123    15

<b>1</b> ФИО работника:	<b>6</b> Единица товара для премии (ед.):
Иванов Петр Князевич	5
<b>2</b> Должность работника:	<b>7</b> НДФЛ (%):
Слесарь	13
<b>3</b> Расценка за единицу продукции:	<b>8</b> Норма (ед.):
Компрессор - 1000 руб.	1
<b>4</b> Изготовленная продукция (ед.):	<b>9</b> Районный коэффициент:
0	Республика Башкортостан
<b>5</b> Премия (руб.):	<b>10</b> Учет МРОТа
1000	Учитывать
<b>11</b> <span style="background-color: blue; color: white; padding: 2px 10px;">Рассчитать</span>	

**16**

УГАТУ 2021 - Курсовая работа - Вариант 1

### Рисунок 2 - Основная страница программы

На основной странице находятся:

1. Поле ввода ФИО работника
2. Поле ввода должности работника
3. Выпадающее меню с выбором продукции
4. Поле ввода, изготовленной работником продукции (в ед.)
5. Поле ввода премии, начисляемой работнику после производства единиц товара до премии.
6. Поле ввода единиц произведенного товара, необходимых работнику до начисления премиальных рублей.
7. Поле ввода текущей ставки НДФЛ в процентах.
8. Поле ввода нормы, сверх которой начисляется премия работнику.
9. Выпадающее меню с выбором районного коэффициента, в котором находится работник
10. Выпадающее меню с выбором учета МРОТа.
11. Кнопка, по нажатии на которую произойдет расчет заработной платы.
12. Выпадающее меню «Файл» с выбором дальнейшей
13. Выпадающее меню «Экспорт», позволяющее работнику экспорттировать результаты в Excel

14. Выпадающее меню «О приложении», позволяющее работнику узнать о приложении

15. Табличка, содержащая логин вошедшего в программу пользователя.

16. Надпись, содержащая дополнительную информацию о программе

Для выполнения расчета заработной платы оператору необходимо заполнить поля пункты с 1 по 10, затем нажать на кнопку «Рассчитать» (11)

Для получения печатной формы необходимо выбрать «Экспорт» (13) и выбрать желаемый из доступных форматов (рис. 3), после чего ожидать загрузки документа.

ФИО работника:	Студен Группы ПИ	Единица товара для премии (ед.):	3
Должность работника:	Слесарь	НФДЛ (%):	13
Расценка за единицу продукции:	Компрессор - 1000 руб.	Норма (ед.):	1
Изготовленная продукция (ед.):	5	Районный коэффициент:	Республика Башкортостан
Премия (руб.):	1254	Учет МРОТа	Учитывать
<b>Рассчитать</b>			

УГАТУ 2021 - Курсовая работа - Вариант 1

**Рисунок 3 - Выбор экспорта в Excel**

Для получения более подробной информации о калькуляторе и о разработчиках оператору следует направится в раздел «О приложении» (14) и выбрать «Справка» или «О разработчиках» соответственно.

Для получения доступа к редактированию продукции или районных коэффициентов оператору необходимо авторизоваться под учетной записью администратора (рис. 4). Где появится раздел «Редактировать». По нажатию на него оператор сможет выбрать пункт «Продукция» или «Коэффициенты» для редактирования продукции (рис. 5) и районных коэффициентов (рис. 6) соответственно.

10  
1304.300001.000 34

Калькулятор Файл Экспорт О приложении Редактировать Администратор admin

ФИО работника:	Иванов Петр Князевич	Единица товара для премии (ед.):	5
Должность работника:	Слесарь	НФДЛ (%):	13
Расценка за единицу продукции:	Компрессор - 1000 руб.	Норма (ед.):	1
Изготовленная продукция (ед.):	0	Районный коэффициент:	Республика Башкортостан
Премия (руб.):	1000	Учет МРОТа	Учитывать

**Рассчитать**

УГАТУ 2021 - Курсовая работа - Вариант 1

Рисунок 4 -Авторизация администратора

Калькулятор Файл Экспорт О приложении Редактировать Администратор admin

ФИО работника:	Иванов Петр Князевич	Единица товара для премии (ед.):	5
Должность работника:	Слесарь	Редактирование	
Расценка за единицу продукции:	Компрессор - 1000 руб.	1000	Удалить
Изготовленная продукция (ед.):	0	300	Удалить
Премия (руб.):	1000	400	Удалить
	Сопло	100	Удалить
	Смеситель	200	Удалить

**Добавить новый** **Закрыть и сохранить** **Рассчитать**

УГАТУ 2021 - Курсовая работа - Вариант 1

Рисунок 5 - Редактирование продукции

**Рисунок 6 - Редактирование районных коэффициентов**

Для того чтобы заново начать расчет заработной платы нужно нажать на «Файл» (12) и выбрать пункт «Новый» (рис. 7). Это сбросит все введенные данные.

Для того чтобы сменить пользователя нужно нажать на «Файл» (12) и выбрать пункт «Выход» (рис. 7). Это перекидывает оператора к окну авторизации.

**Рисунок 7 - Начало расчетов с начала и возврат к окну авторизации**

#### 4. Сообщения оператору

При регистрации, если аккаунт с введенными данными может быть создан, выводится сообщение об успешном выполнении операции (рис. 8).

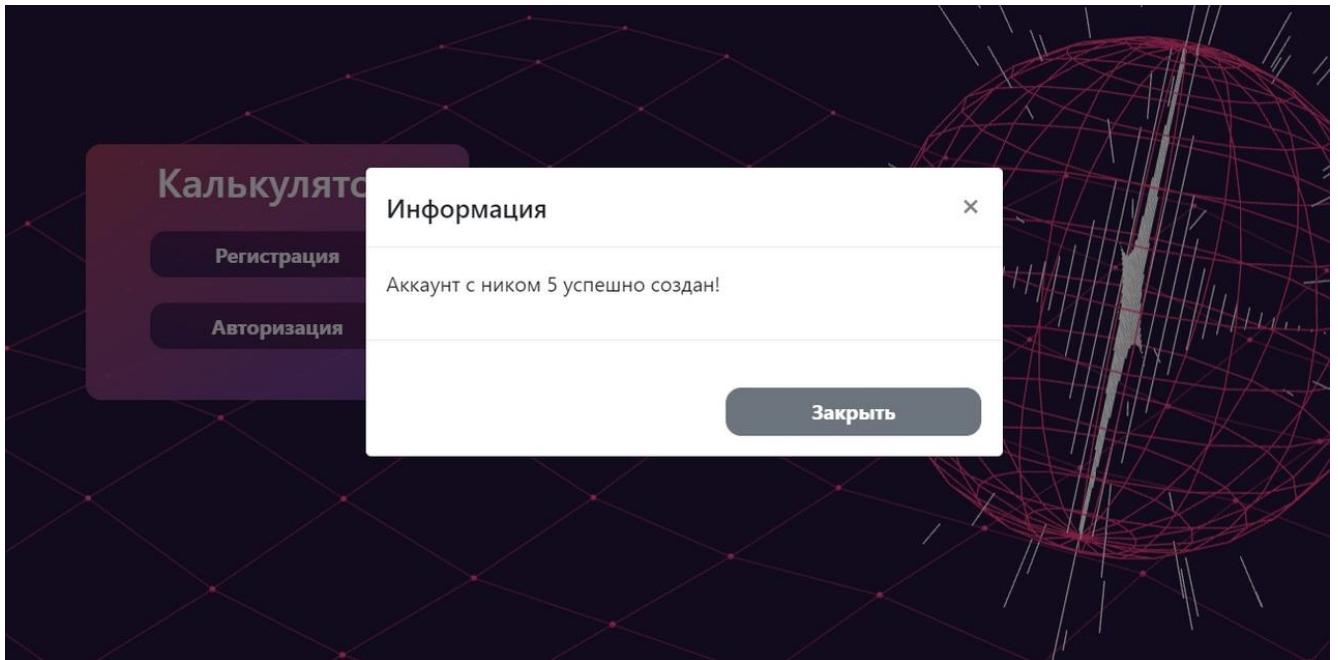


Рисунок 8 - Сообщение об успешной регистрации нового пользователя

При авторизации, если аккаунт с введенными данными не существует, то есть неверно введены логин и/или пароль - выводится сообщение о провале авторизации (рис. 9).

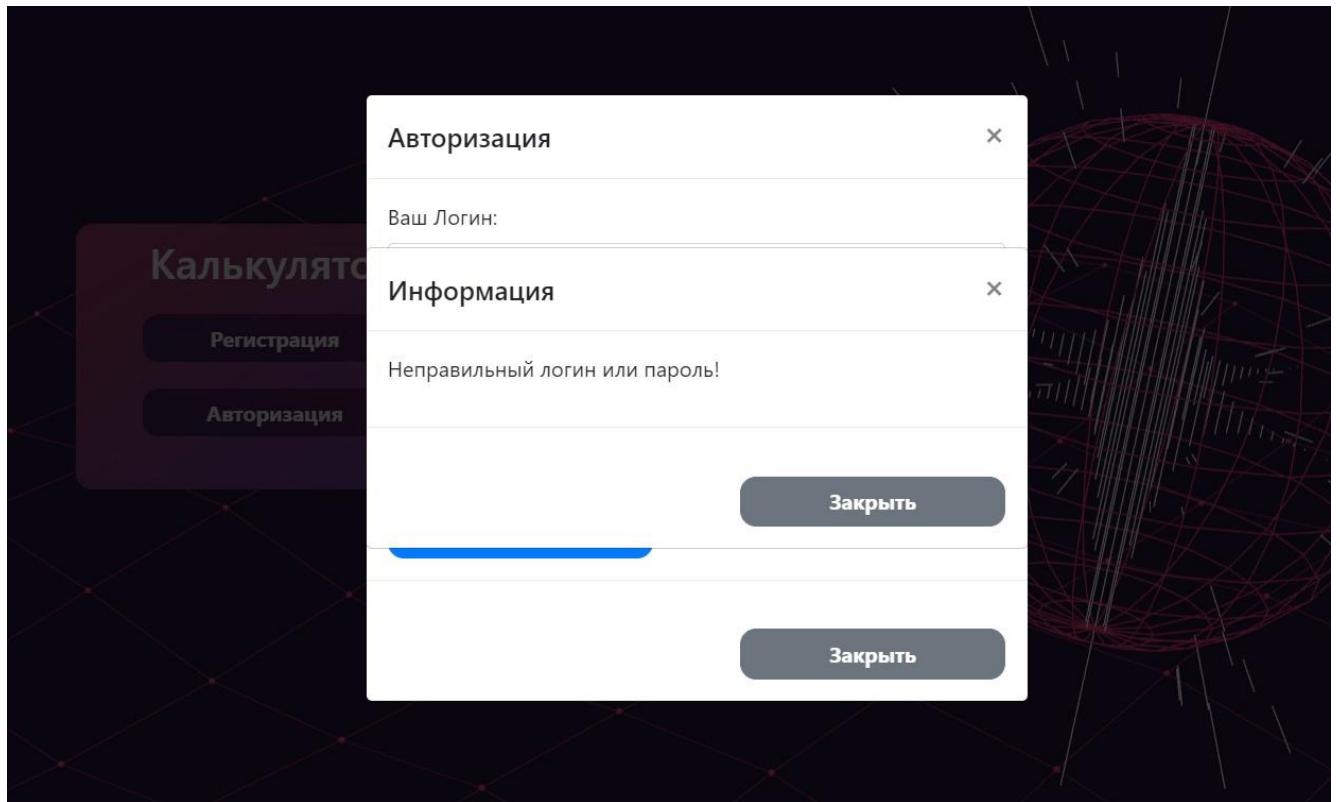


Рисунок 9 - Сообщение о неуспешной авторизации пользователя

При регистрации, если введенные пароли отличаются, то выводится сообщение, предупреждающее об этом (рис. 10).

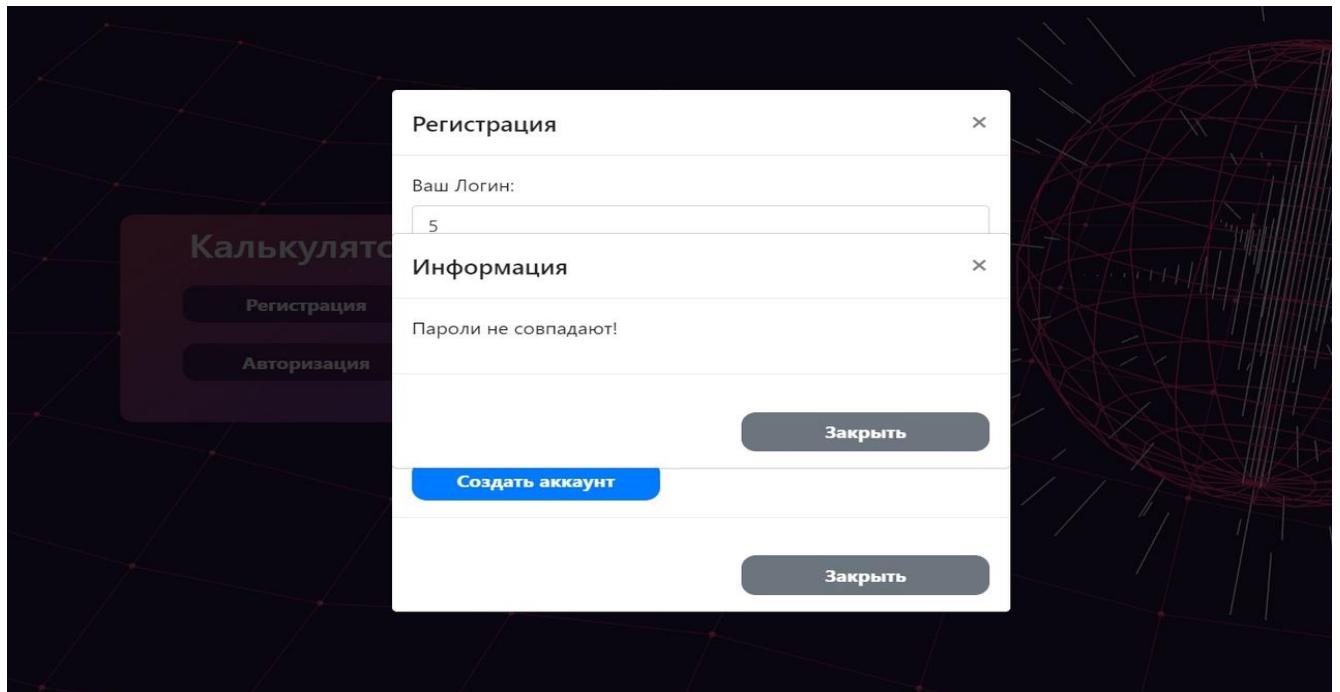


Рисунок 10 - Сообщение о неуспешной регистрации при вводе разных паролей

При регистрации, если аккаунт с введенным логином уже существует, то выводится сообщение, предупреждающее об этом (рис. 11).

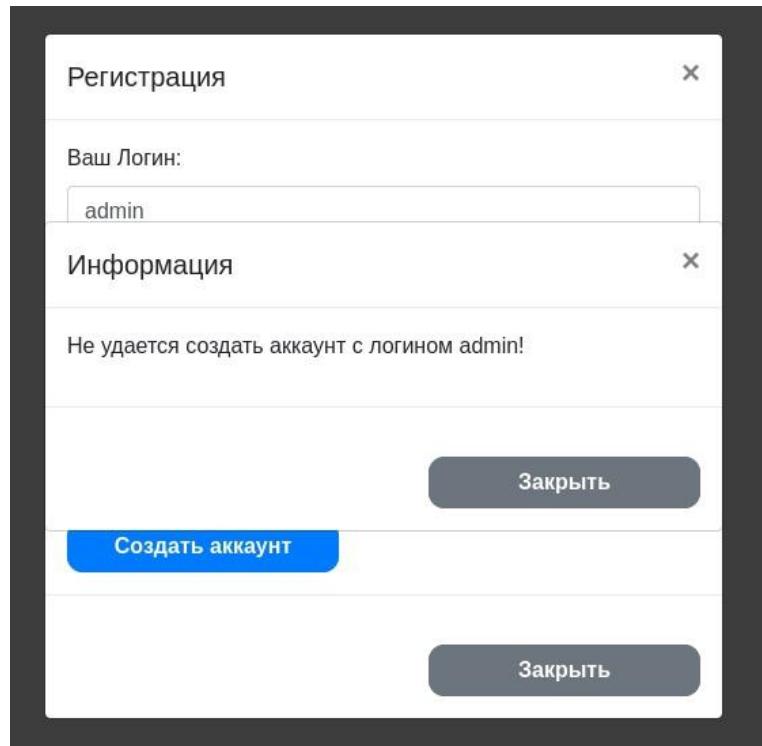


Рисунок 11 - Сообщение о неудачной регистрации при повторном использовании логина

По нажатию «Рассчитать», если все поля были заполнены корректно - выводится сообщение с результатом расчета заработной платы (рис. 12)

The screenshot shows a salary calculation interface. At the top right is a red box labeled 'Администратор admin'. The form fields include: 'ФИО работника:' (Ivanov Ivan Ivanovich), 'Единица товара для премии (ед.):' (5), 'Должность работника:' (Слесарь), 'НФДЛ (%):' (13), 'Расценка за единицу продукции:' (Компрессор - 1000 руб.), 'Норма (ед.):' (15), 'Изготовленная продукция (ед.):' (20), and 'Премия (руб.):' (1000). A central modal window titled 'Результат вычисления' (Calculation result) displays the message 'Заработка плата Иванов Иван Иванович составит 21010.499999999996 у.е.' (The salary of Ivanov Ivan Ivanovich will be 21010.499999999996 rubles). At the bottom right of the modal is a 'Закрыть' (Close) button, and at the bottom right of the main interface is a blue 'Рассчитать' (Calculate) button.

Рисунок 12- Сообщение с результатом расчета заработной платы

По нажатию «Рассчитать», если не все поля были заполнены корректно - выводится сообщение предупреждающее об этом (рис. 13).

The screenshot shows a web-based calculator application. At the top right, it says "Администратор admin". The main form fields include:

- ФИО работника: Иванов Иван Иванович
- Единица товара для премии (ед.): 5
- Должность работника: Слесарь
- НФДЛ (%): 13
- Расценка за единицу продукции: Компрессор - 1000 руб.
- Норма (ед.): 15
- Изготовленная продукция (ед.): 20
- Премия (руб.): фывафыв

A modal window titled "Ошибка" (Error) is displayed in the center, containing the message: "Пожалуйста, заполните все поля корректными данными." (Please fill all fields with correct data). There are "Закрыть" (Close) and "Рассчитать" (Calculate) buttons at the bottom of the modal.

Рисунок 13 - Сообщение о неверном вводе данных для расчета заработной платы

При выборе пункта «О приложении» затем «Справка» показывается справка с информацией о Калькуляторе (рис. 14).

The screenshot shows the same calculator application interface as in Figure 13. The main form fields are identical. A modal window titled "Справка" (Help) is displayed in the center, containing the following text:

Данный калькулятор предназначен для расчёта сдельно-премиальной зарплаты.  
Сдельно-премиальная оплата труда – это форма, которая предполагает не только исчисление прямого заработка с учетом фактических результатов труда, но и установление надбавок (премий) за выполнение и перевыполнение плановых показателей.  
Для расчёта необходимо ввести все значения.  
Расчёт осуществляется с учётом МРОТ, НДФЛ.

At the bottom of the modal, there are "Закрыть" (Close) and "Рассчитать" (Calculate) buttons.

Рисунок 14 - Сообщение со справкой о приложении

При выборе пункта «О приложении», затем «Разработчик» показывается информация о разработчиках (рис. 15).

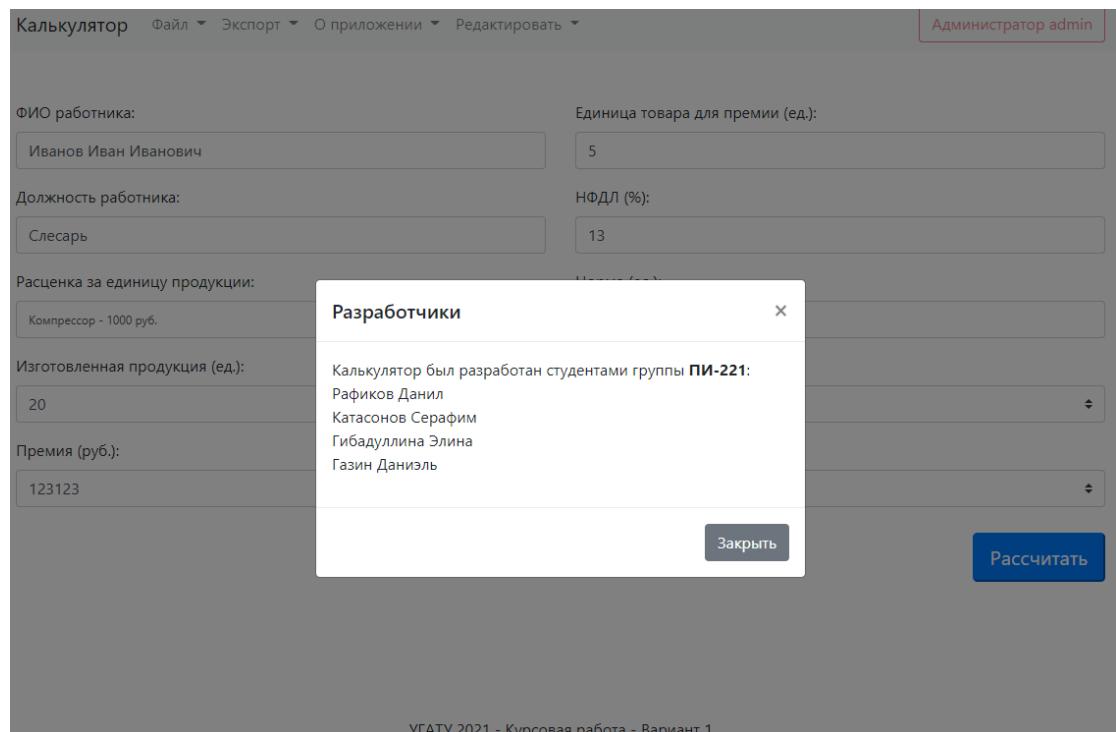


Рисунок 15 - Сообщение с информацией о разработчиках

**ПРИЛОЖЕНИЕ 8**

**ФГБОУ ВО УФИМСКИЙ ГОСУДАРСТВЕННЫЙ АВИАЦИОННЫЙ  
ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**СОГЛАСОВАНО**

Доцент кафедры АСУ ФГБОУ  
ВО УГАТУ

Личная  
подпись

24.05.2021

Расшифровка  
подписи

**УТВЕРЖДАЮ**

Студент группы ПИ-221  
ФИРТ ФГБОУ УГАТУ, модератор  
Личная  
подпись

24.05.2021

Расшифровка  
подписи

**Калькулятор сделко-премиальной зарплаты**

**Руководство программиста**

**ЛИСТ УТВЕРЖДЕНИЯ**

**1304.300001.000 33-ЛУ**

**СОГЛАСОВАНО**

Представитель  
команды разработчиков

Доцент кафедры АСУ  
ФГБОУ ВО УГАТУ

Личная  
подпись

24.05.2021

Расшифровка  
подписи

Студент группы ПИ-221

ФИРТ ФГБОУ ВО УГАТУ, модератор

Личная  
подпись

Расшифровка  
подписи

24.05.2021

ФИО. Н. ИОАН.	Полн. И. ИАНА	Личн. И. ИАНА	Узак. И. ИАНА	Узак. И. ИАБЛ.	Полн. И. ИАНА

2021

Утвержден  
1304.300001.000 33-ЛУ

ФГБОУ ВО УФИМСКИЙ ГОСУДАРСТВЕННЫЙ АВИАЦИОННЫЙ  
ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

**Программный продукт  
«Калькулятор сдельно-премиальной зарплаты»  
Руководство программиста  
1304.300001.000 34  
Листов 12**

Имя, Ф. И. О.	Полиг. № листа	Взам. лист. №	Изл. № кубл.	Полиг. № листа

## АННОТАЦИЯ

Данный документ является руководством программиста для программного продукта «Калькулятор сдельно-премиальной зарплаты». Документ разработан в рамках курсового проекта в соответствии с ГОСТ 19.504-79. Единая система программной документации. Руководство программиста. Требования к содержанию и оформлению.

## **СОДЕРЖАНИЕ**

1. Назначение и условия применения программы.....	5
2. Характеристики программы.....	7
3. Обращение к программе.....	8
4. Входные и выходные данные.....	9
5. Сообщения.....	10

## **1. Назначение и условия применения программы**

### **1.1. Назначение программы**

Функциональное назначение программного продукта: программа предназначена для автоматизации расчета заработной платы для сотрудника-пользователя и возможность изменения настроек для администратора.

Эксплуатационное назначение: сотрудники, работающие в структурном подразделении «Цех», например, рабочий.

Данный калькулятор предназначен для предприятия ООО «Авиастрой».

### **1.2. Функции, выполняемые программой**

Для программы реализована многопользовательская работа. Поэтому в программном продукте есть модуль для авторизации пользователей, выделяющий две категории пользователей: простой пользователь и администратор.

Для авторизации необходимо заполнить два поля: логин и пароль.

Также у пользователей есть возможность регистрации.

Для простого пользователя предусмотрены следующие возможности:

- Ввести данные для расчета заработной платы;
- Получить результат расчёта;
- Сформировать печатную форму в формат .xls с результатами.

Для администратора есть возможность изменения настроек главной формы пользователя. Он способен изменять расценку на товары и значение районных коэффициентов.

### **1.3. Условия, необходимые для выполнения программы**

Требования к серверу, необходимые для функционирования веб-приложения:

- процессор, ОЗУ и видеокарта (интегрированная или внешняя)
- должны позволять запустить операционную систему Windows,Linux Ubuntu или Linux Mint;
- Поддерживаемые протоколы передачи данных: HTTP / HTTPS;
- Количество подключенных пользователей 250;
- Процессор 4 ядра, тактовая частота 2.90 ГГц и выше;
- Платформа 32-х или 64-х разрядная;
- Оперативная память- 10 ГБ и выше;
- Жесткий диск- 300 МБ свободного объема и выше.
- Поддержка языка программирования Java, поддержка хранения и обработки данных;
- Интеграция с github;
- Поддержка Apache http.

## 2. Характеристики программы

Для работы с Калькулятором не требуется установка дополнительного программного обеспечения.

Пользователь работает с программой через веб-браузер, где у него есть доступ к веб-приложению, расположенному по определённому url-адресу: <https://team1-calculator.herokuapp.com/>.

Для работы с Калькулятором у пользователя должна быть любая из следующих операционных систем и установлен браузер:

- Windows 10, а именно браузеры: Edge, Google Chrome, Mozilla Firefox;
- Linux Ubuntu - браузеры Mozilla Firefox, Google Chrome;
- Linux Mint - браузеры Mozilla Firefox, Google Chrome.

### 3. Обращение к программе

Запуск приложения возможен при переходе на сайт: <https://pi221team1-course.herokuapp.com/>. После перехода по ссылке, откроется окно авторизации/регистрации. Пользователю необходимо зарегистрироваться (если у него нет аккаунта в системе), или авторизоваться (если у него есть аккаунт в системе) для перехода в основное окно с расчетами.

Внутренне управление программным продуктом производится разработчиками, которые имеют доступ к репозиториоу ( ссылка на репозиторий: <https://github.com/Gibadullina/calculator-team1>) и может вносить свои изменения. Данные изменения запускают автоматическую сборку проекта и развертывание его через *Heroku*.

#### **4. Входные и выходные данные**

Входные данные:

Данные, введенные пользователем на поля формы для входа (пароль, логин), для редактирования продукции или районных коэффициентов в панели администратора, для расчета заработной платы (ФИО работника, должность работника, изготовленная продукция, премия, единицы товара для премии, ставка НДФЛ, норма ед.) и выбранные из выпадающего меня (Расценка за единицу продукции, районный коэффициент, учет МРОТ).

Выходные данные:

Вывод всплывающим окном результатов вычислений заработной платы или их экспорт в файл формата .xls.

## 5. Сообщения

В данном разделе указаны, выдаваемые обычному пользователю или администратору при выполнении действий.

Сообщение, выдаваемое в случае неправильного ввода пароля или логина при авторизации (рис.1).

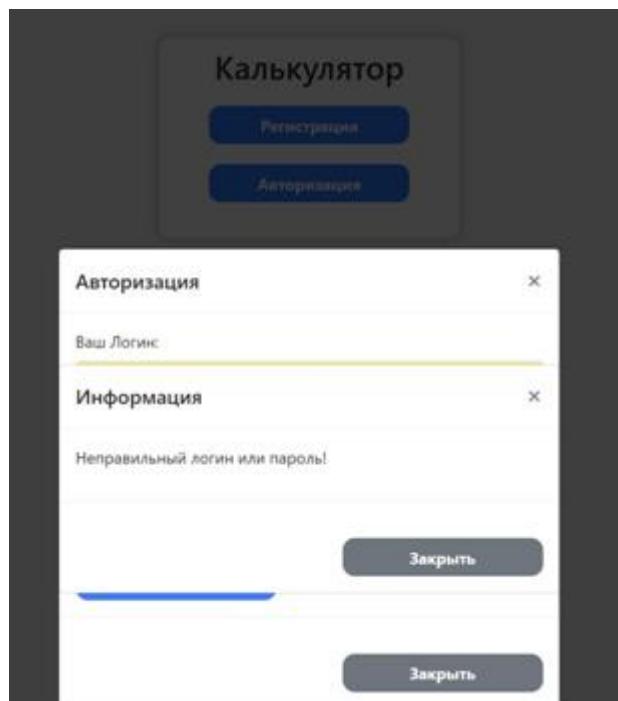


Рисунок 1 - Сообщение о неверном пароле/логине

Данное сообщение необходимо закрыть, нажав кнопку. И попытаться ввести свои данные снова, проверив, не нажата ли клавиша CapsLock, та ли стоит раскладка клавиатуры.

При регистрации нового пользователя могут быть выданы три вида сообщений.

При успешной регистрации (рис.2), необходимо закрыть сообщение и далее авторизоваться, введя логин и пароль.

11  
1304.300001.000 33

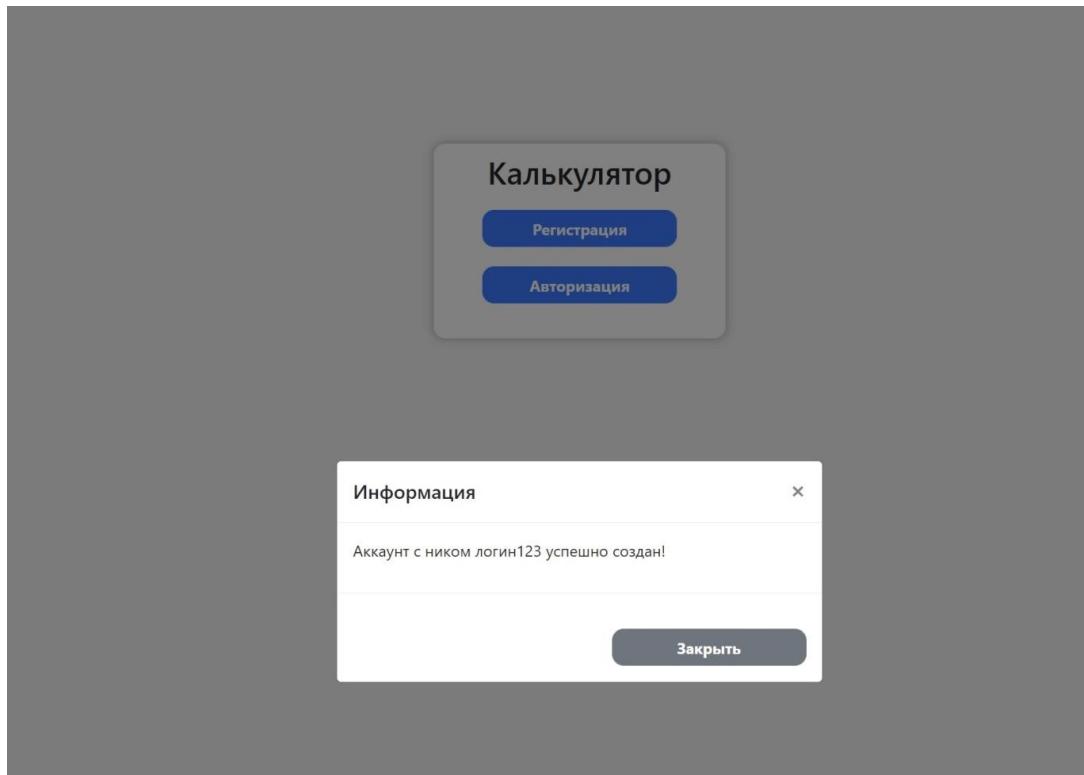


Рисунок 2- Успешная регистрация

При регистрации, если введенные пароли отличаются, то выводится сообщение, предупреждающее об этом (рис.3). Необходимо также закрыть данное сообщение и попробовать снова.

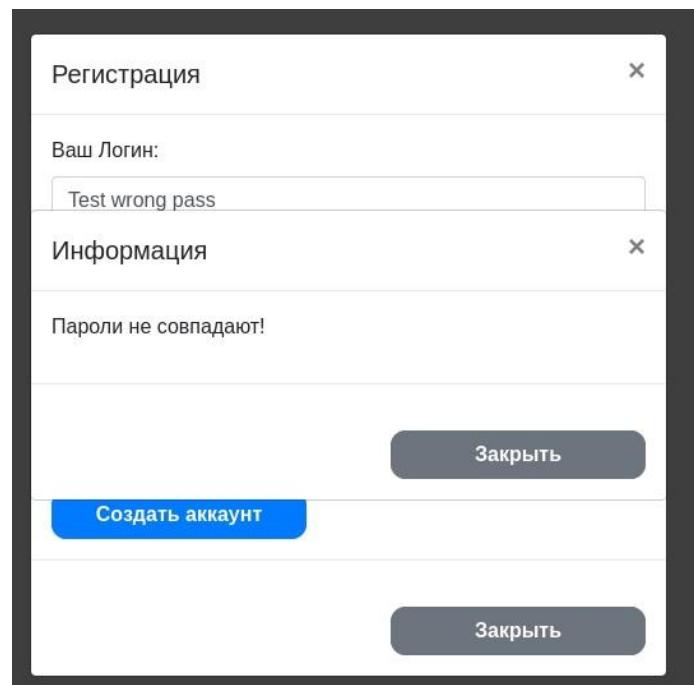


Рисунок 3- Сообщение о неудачной регистрации при вводе разных паролей

При регистрации, если аккаунт с введенным логином уже существует, то выводится сообщение, предупреждающее об этом (рис.4). Закройте сообщение и попробуйте снова.

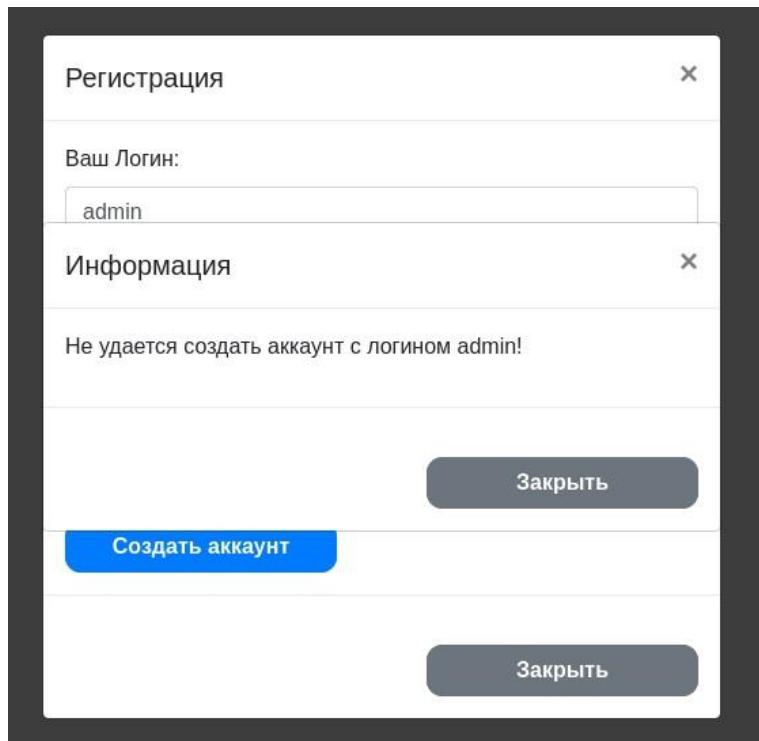


Рисунок 4- Сообщение о неудачной регистрации при повторном использовании логина

По нажатию «Рассчитать», на странице с расчетом, если не все поля были заполнены корректно - выводится сообщение предупреждающее об этом (рис.5). Закройте сообщение и заполните поля, корректными значениями, которые должны соответствовать наименованию полей ввода.

13  
1304.300001.000 33

Калькулятор    Файл ▾    Экспорт ▾    О приложении ▾    Редактировать ▾    Администратор admin

ФИО работника: Иванов Иван Иванович    Единица товара для премии (ед.): 5

Должность работника: Слесарь    НФДЛ (%): 13

Расценка за единицу продукции: Компрессор - 1000 руб.    Норма (ед.): 15

Изготовленная продукция (ед.): 20

Премия (руб.): фывафыв

**Ошибка**  
Пожалуйста, заполните все поля корректными данными.

**Закрыть**    **Рассчитать**

УГАТУ 2021 - Курсовая работа - Вариант 1

Рисунок 5 - Сообщение о неверном вводе данных для расчета заработной платы

## **СПИСОК ЛИТЕРАТУРЫ**

1. Трудовой кодекс Российской Федерации от 30 декабря 2001 г. № 197-ФЗ (с изм. от 9 ноября 2020 г.) // Собрание законодательства РФ. 2002.
2. Сдельная оплата труда [Электронный ресурс] // Audit-it.ru: сайт. — URL: [https://www.audit-it.ru/terms/trud/sdelnaya\\_oplata\\_truda.html](https://www.audit-it.ru/terms/trud/sdelnaya_oplata_truda.html).
3. Кучина, Ю.А Трудовое право 3-е изд., пер. и доп. Учебник для академического бакалавриата [Текст] / Ю. А Кучина, С. Ю. Головина, 2016 - 1256 с.
4. Федеральный закон от 19.06.2000 № 82-ФЗ (ред. от 27.12.2019) "О минимальном размере оплаты труда".
5. Федеральный закон от 12 января 1996 г. N 10-ФЗ "О профессиональных союзах, их правах и гарантиях деятельности" (ред. От 8 декабря 2020 г.)
6. Налоговый кодекс Российской Федерации. Часть вторая от 5 августа 2000 г. № 117-ФЗ (ред. от 23 ноября 2020 г.).
7. ГОСТ 19.201—78. Единая система программной документации. Техническое задание. Требования к содержанию и оформлению.
8. ГОСТ 19.504—79. Единая система программной документации. Руководство программиста. Требования к содержанию и оформлению.
9. ГОСТ 19.505—79. Единая система программной документации. Руководство оператора. Требования к содержанию и оформлению.
10. ГОСТ 19.404—79. Единая система программной документации. Пояснительная записка. Требования к содержанию и оформлению.