**Министерство науки и высшего образования Российской Федерации**

Федеральное государственное автономное образовательное учреждение высшего образования «Уральский федеральный университет имени первого Президента России Б.Н. Ельцина

Институт радиоэлектроники и информационных технологий – РТФ

Центр ускоренного обучения

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №5**

**По дисциплине «Технологии программирования»**

Студент группы РИЗ-240938у Бабенко М.А.

Преподаватель: Архипов Н.А.

**Екатеринбург, 2025 г.**

**Лабораторная работа N4**

Цель: доработать простой REST сервис и добавить unit тесты.

**Задания для самостоятельной работы**

**Ссылка на GitHub: https://github.com/sverdlovsky00/SpringB\_LW.git**

**Задание 1**

Для начала, изменим Request так, чтобы он соответствовал задаче задания.

Новый код класса Request показан на рисунке 1.

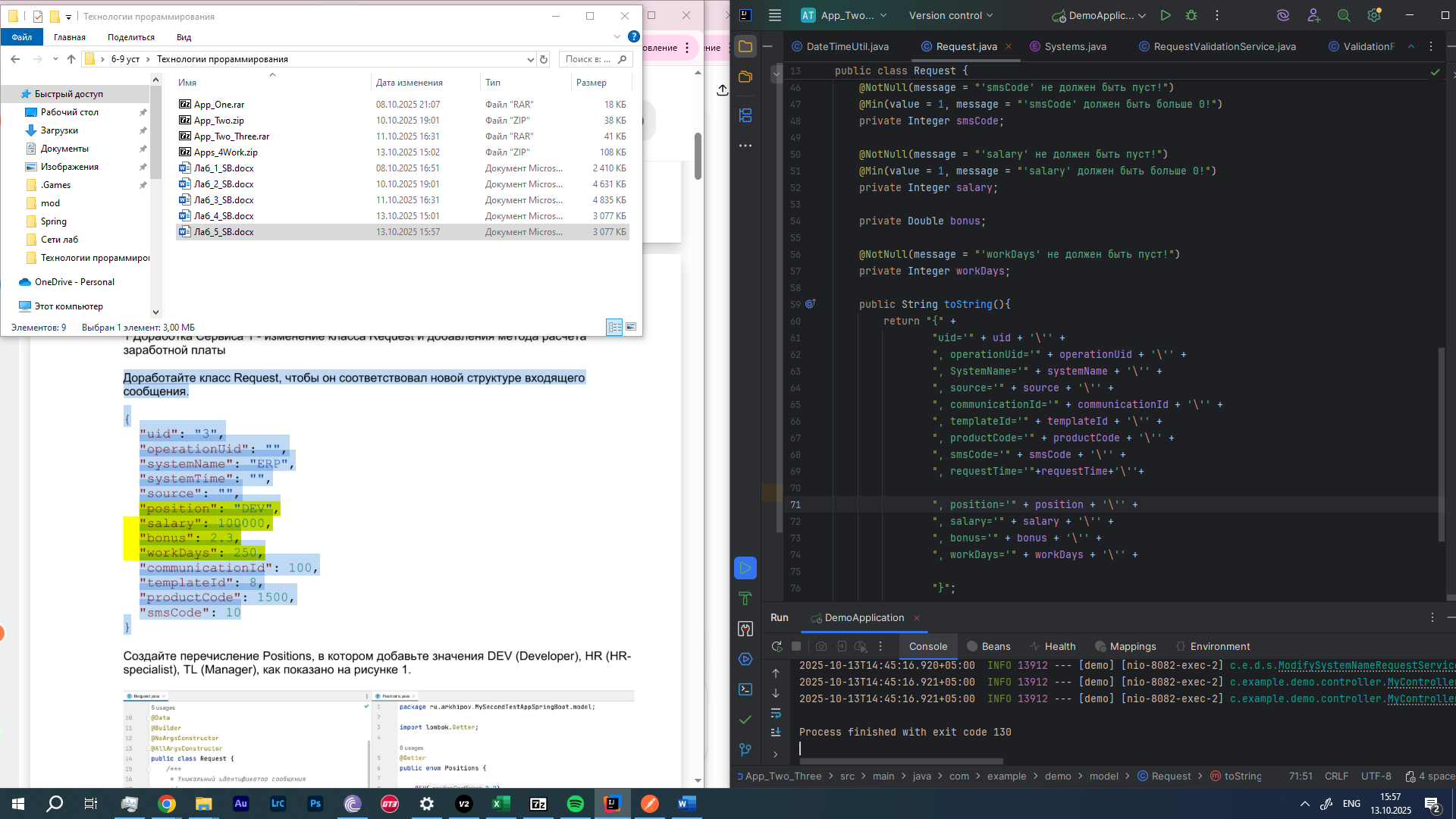


Рисунок 1 – Новый код Request.java

Затем, создаем новый класс – набор констант Positions, показан на рисунке 2.

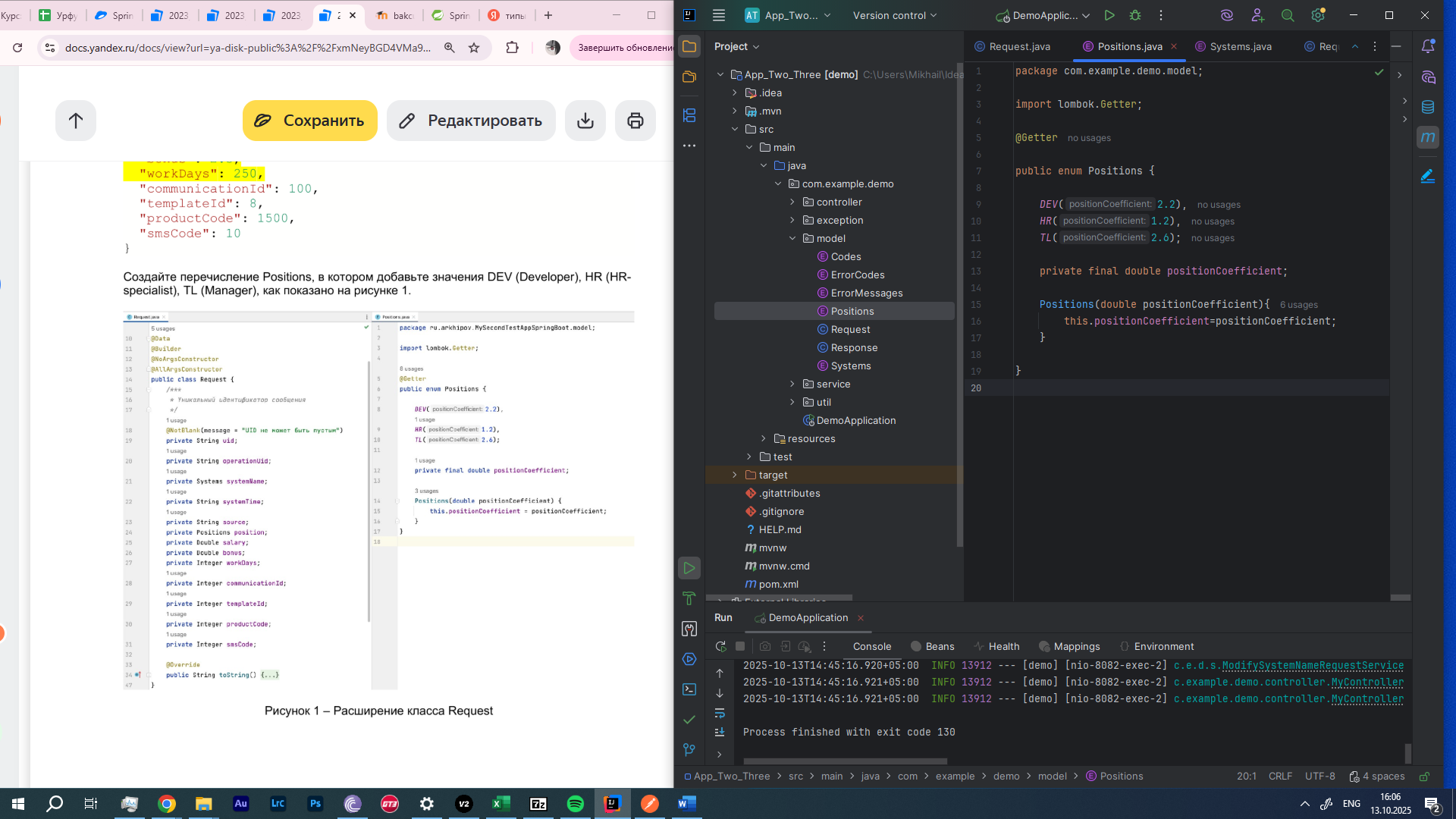


Рисунок 2 – Созданный набор констант Positions

Затем, добавляем строку в Response и создаем в Model два класса, один из которых – интерфейс для другого – AnnualBonusService и AnnualBonusServiceImpl.

Код созданных классов показан на рисунке 3.

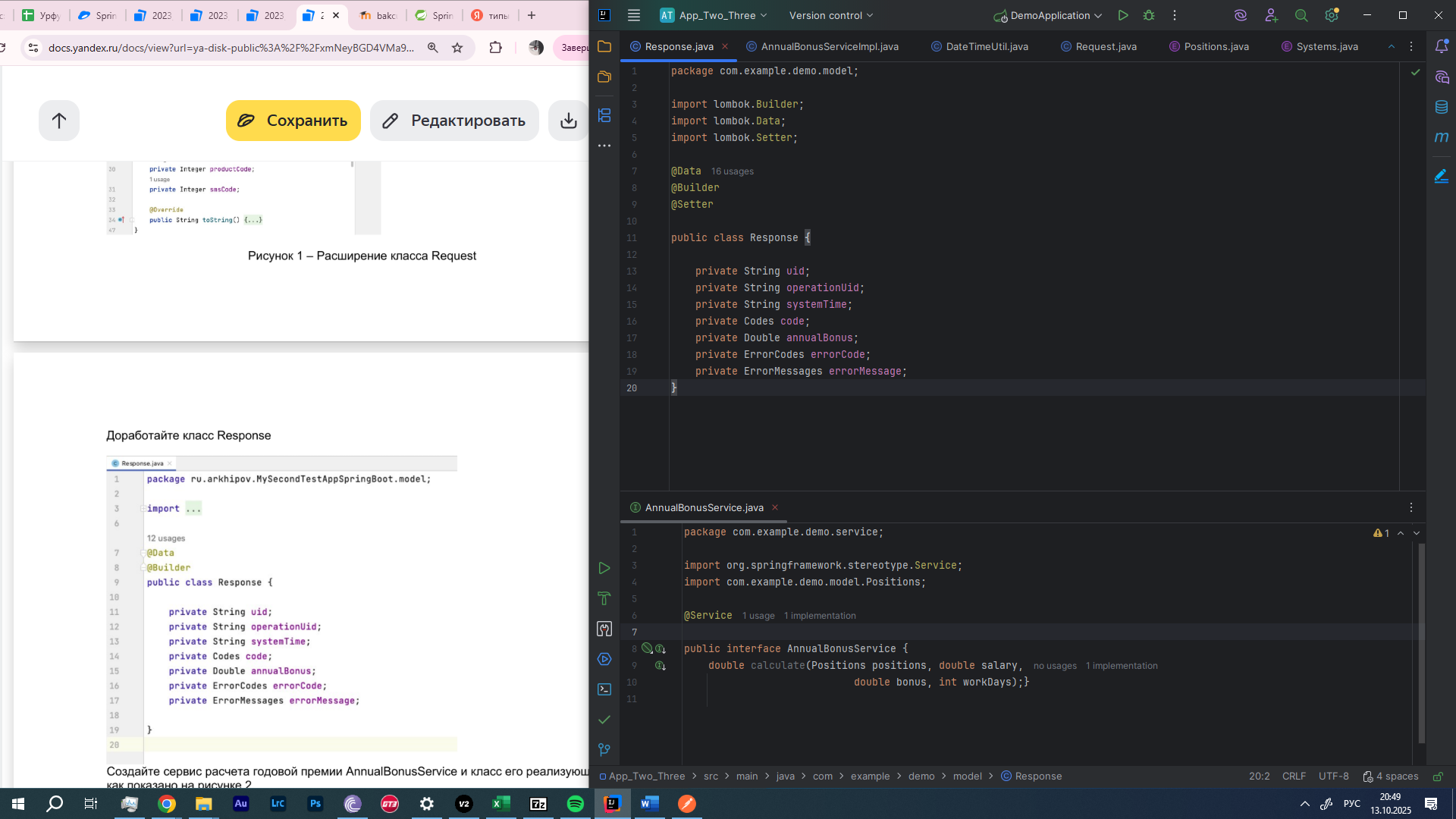


Рисунок 3 – Созданные AnnualBonusService и AnnualBonusServiceImpl

В AnnualBonusServiceImpl выбираем название класса, и для него функцию «Create Test». Создалась «заготовка» нашего теста, код которой мы дополним. Окно опций создания теста показано на рисунке 4.

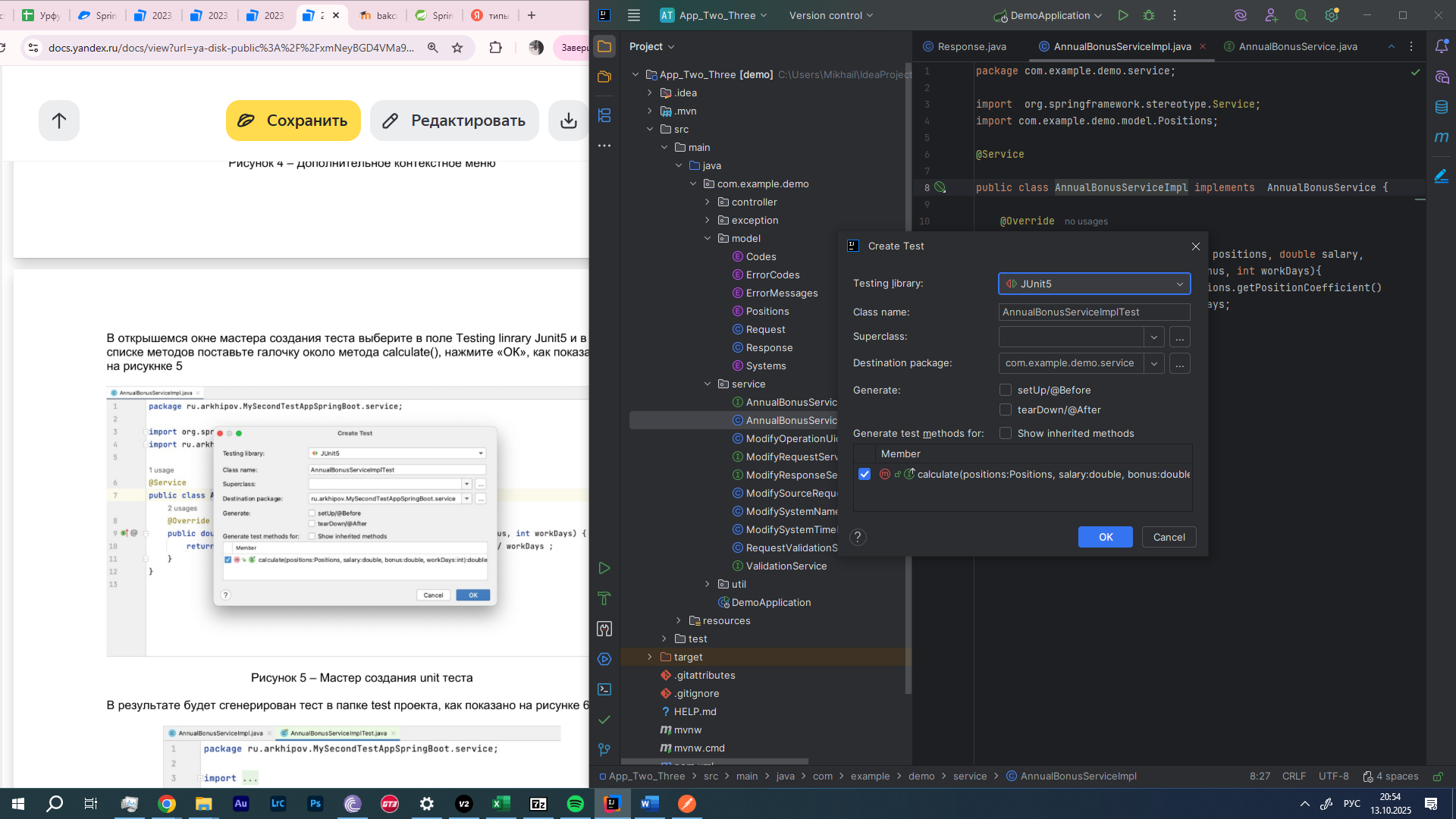


Рисунок 4 – Результат внедрения ModifyRequestService в MyController

Созданная программа тестирования показана на рисунке 5.

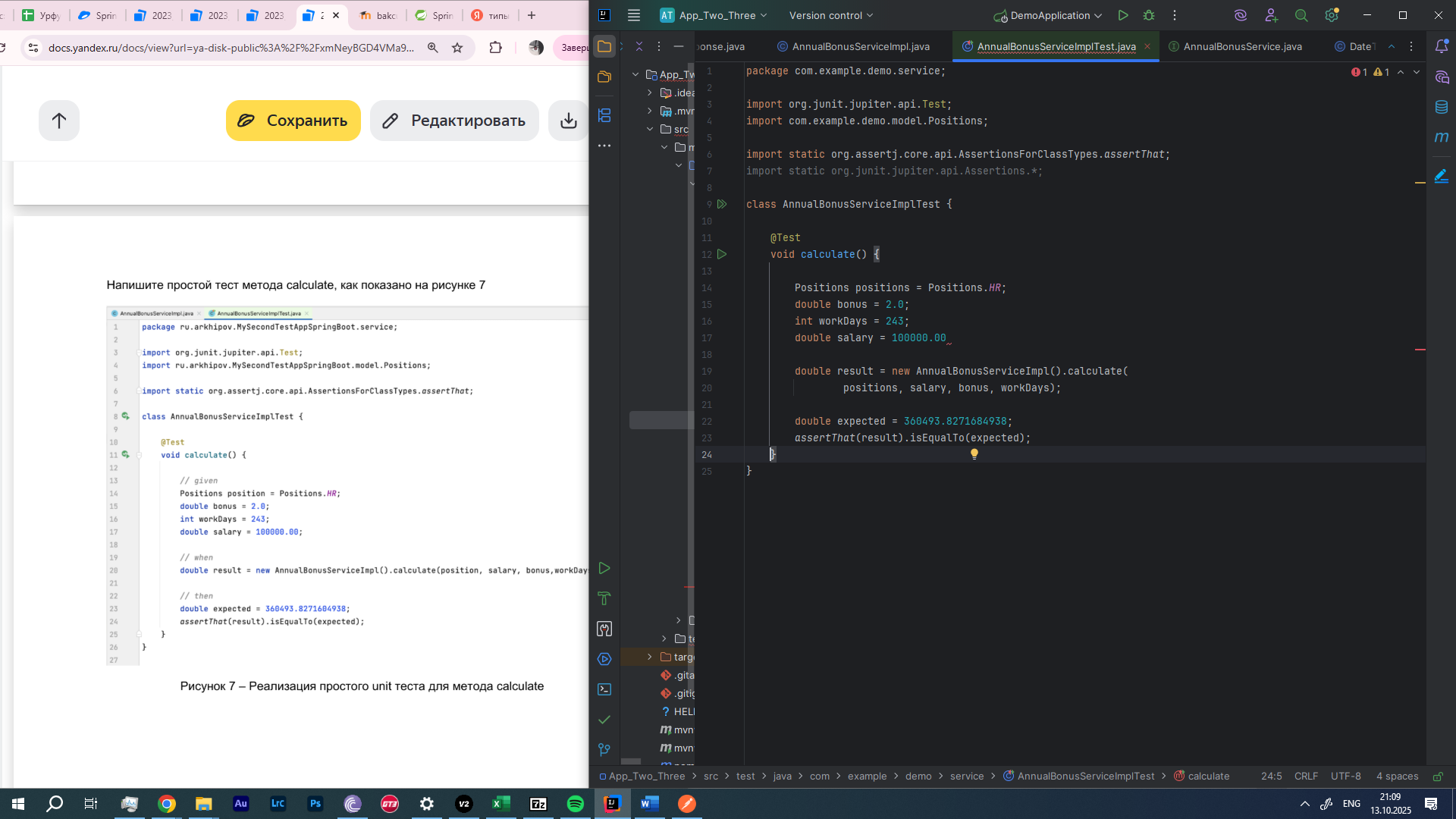


Рисунок 5 – Созданный тест AnnualBonusServiceImplTest

**Задание 2. Реализация дополнительного функционала.**

1 Доработайте метод calculate, чтобы в нем вычислялось количество дней в году, в зависимости от того високосный год или нет.

2 Расширьте enum Positions, добавьте три дополнительных позиции, а также добавьте поле isManager с типом данных boolean, для менеджеров установите значение true, в прочих случаях false.

3 Напишите метод, вычисляющий квартальную премию, метод работает только для менеджеров и иных управленцев (те, у кого isManager == true), добавьте проверку в метод, на признак управленец это или нет.

4 Разработайте unit тест для метода расчета квартальной премии.

5 Произведите рефакторинг приложения – добавьте комментарии к каждому полю в классах Request и Response в соответствии с описанием из лабораторной работы №2, пример комментария к полю показан на рисунке 1 текущей работы.

6 Сделайте так, чтобы в классе MyController стало меньше кода, т.е. приберитесь в

нем.

Для начала, доработаем метод calculate в AnnualBonusServiceImpl. Для того, чтобы определить, количество дней в году – используем встроенный инструмент – метод Year.now().length(), и, при расчетах, умножение будет происходить на эту величину. Новый код метода показан на рисунке 6.

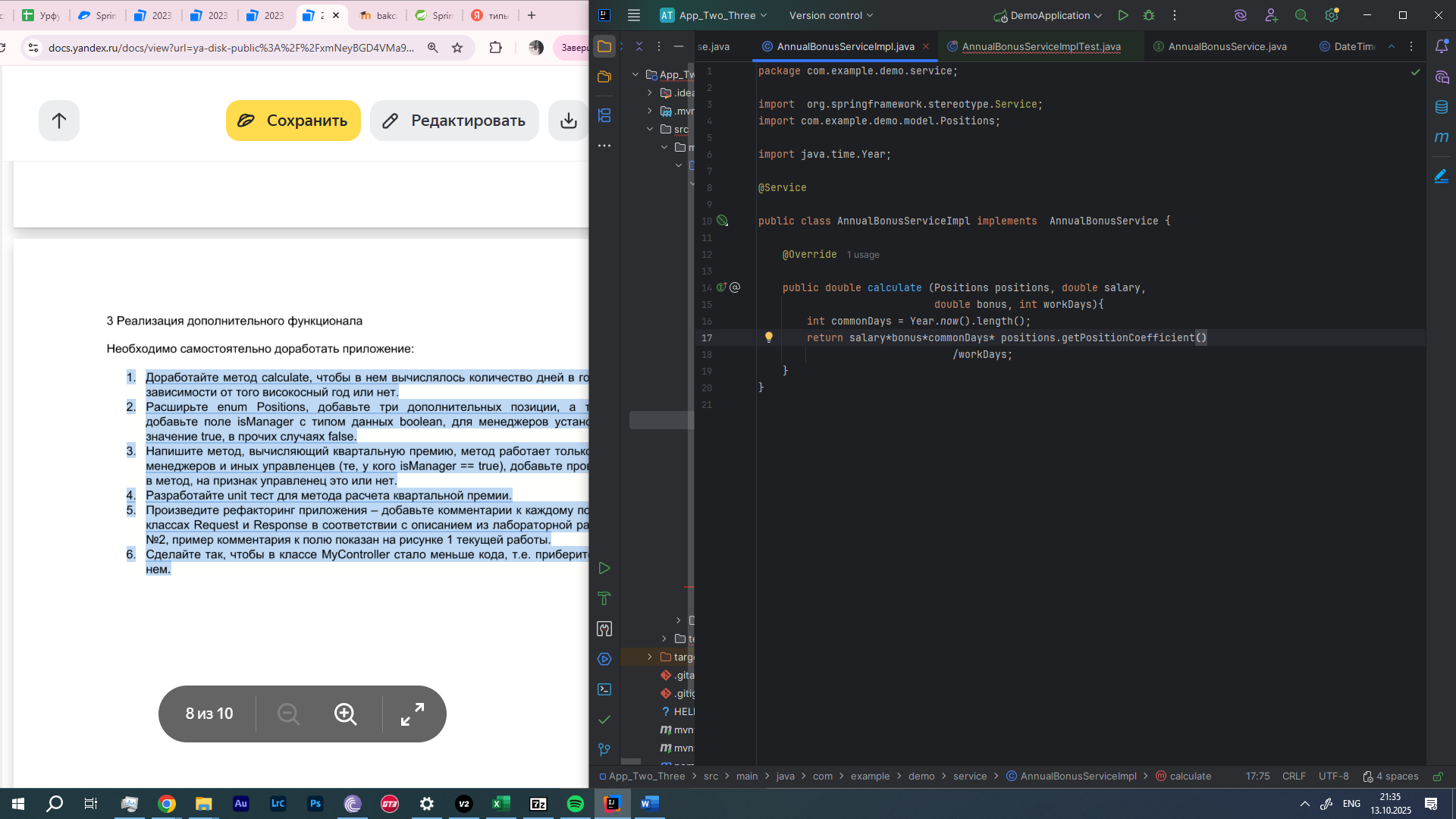


Рисунок 6 – Обновленный AnnualBonusServiceImpl

Теперь нам нужно доработать Positions – добавляем новые три «должности», и новый флаг к ним типа boolean; изменяем конструктор. Обновленный Positions показан на рисунке 7.

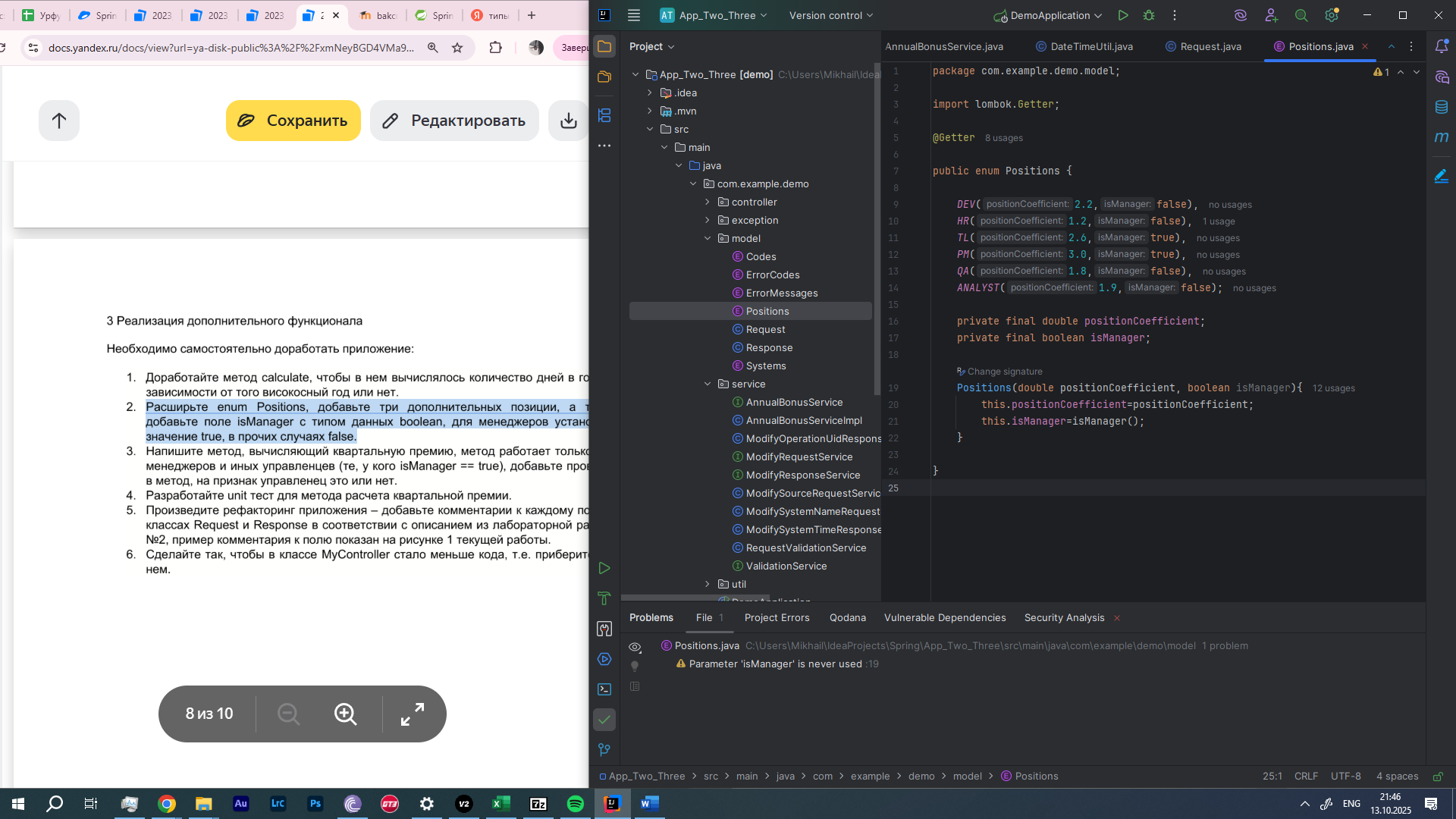


Рисунок 7 – Обновленный Positions

Теперь нам нужен метод, вычисляющий премию. В сервис AnnualBonusServiceImpl добавляется метод calculateQuarterlyBonus. В нем определяется, кому доступна премия (проверка по должности и коэф. Эффективности) с выводом этой информации, а также в результате выполнения возвращается величина рассчитанной премии премии. Дополним вместе с этим интерфейс сервиса – добавим строчку: double calculateQuarterlyBonus(Positions positions, double salary, double quarterPerformance);. Обновленный код показан на рисунке 8.

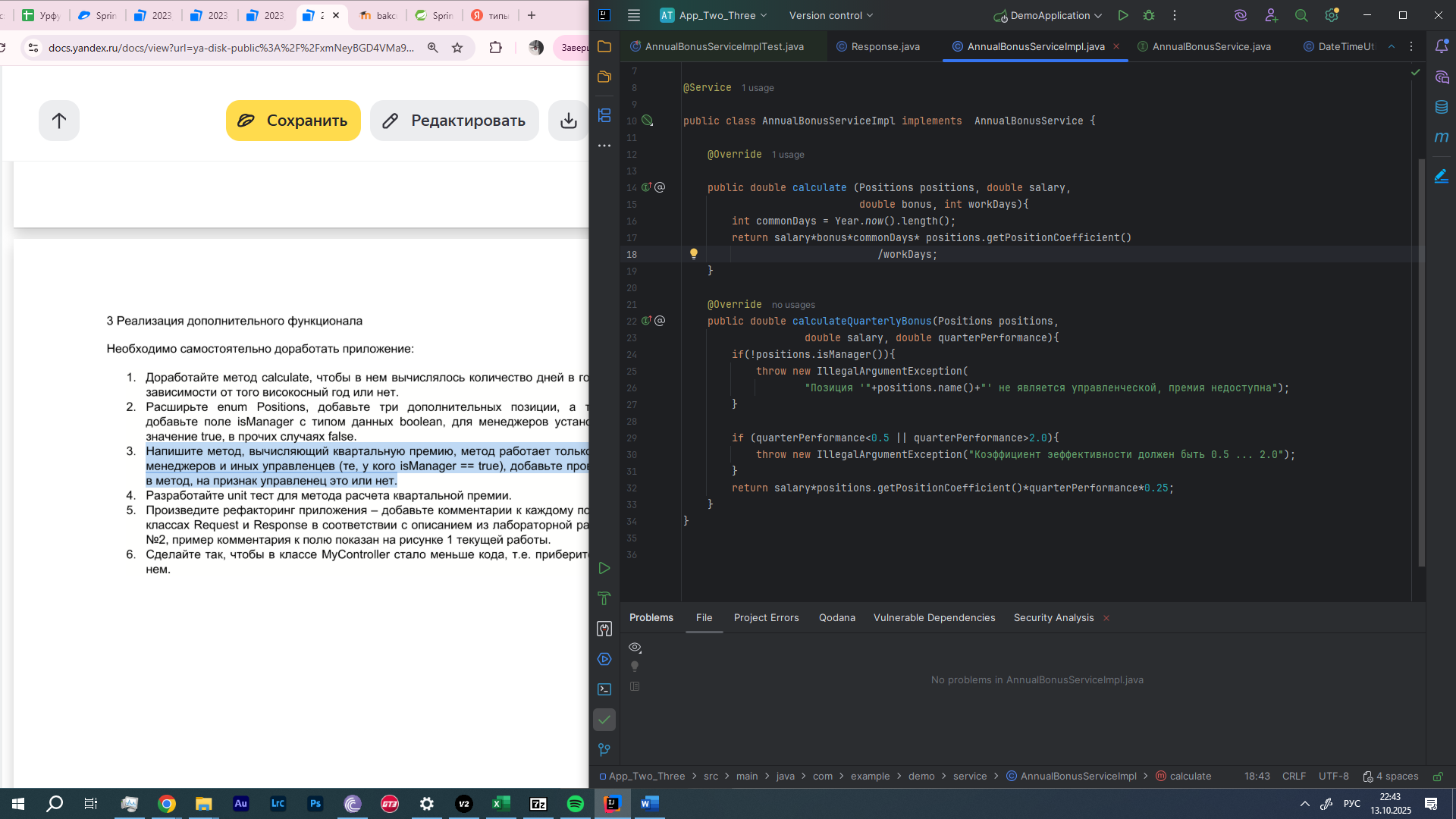


Рисунок 8 – Обновленный код AnnualBonusServiceImpl

Для тестирования будем использовать уже созданный файл с тестом, только его модернизируем для новых задач. Итак, написанный нами тест уже представляет собой расчет общей годовой премии. Нам же нужно создать расчет премии за квартал, и проверить возможные кейсы ошибок: должность с флагом isManager==false, эффективностью ниже порога (0.4), нулевой зарплатой (salary==0). Обновленный тест AnnualBonusServiceImplTest показан на рисунке 9.

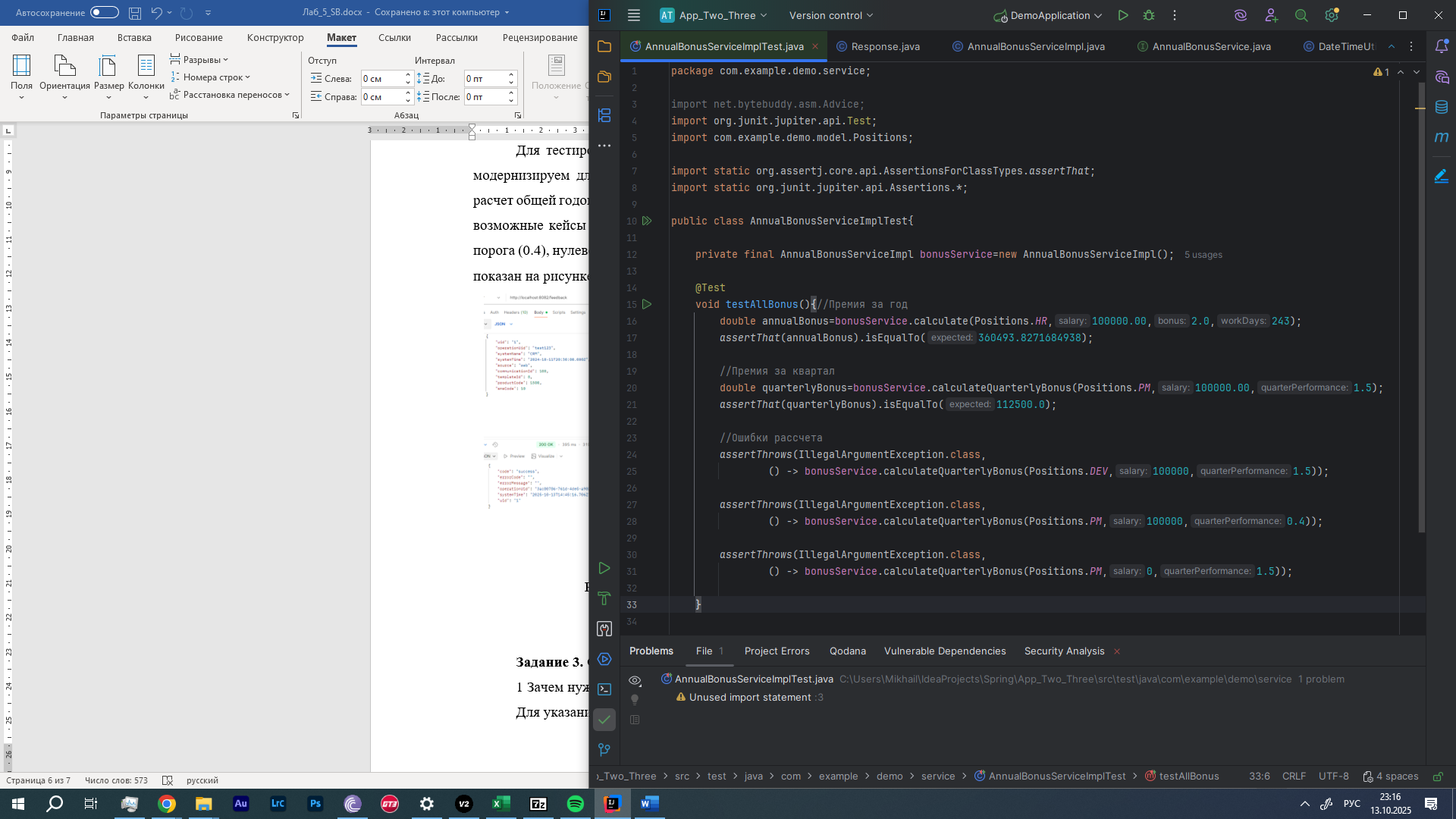


Рисунок 9 – Обновленный AnnualBonusServiceImplTest

Также нам требуется расставить комментарии в Request, процесс показан на рисунке 10.

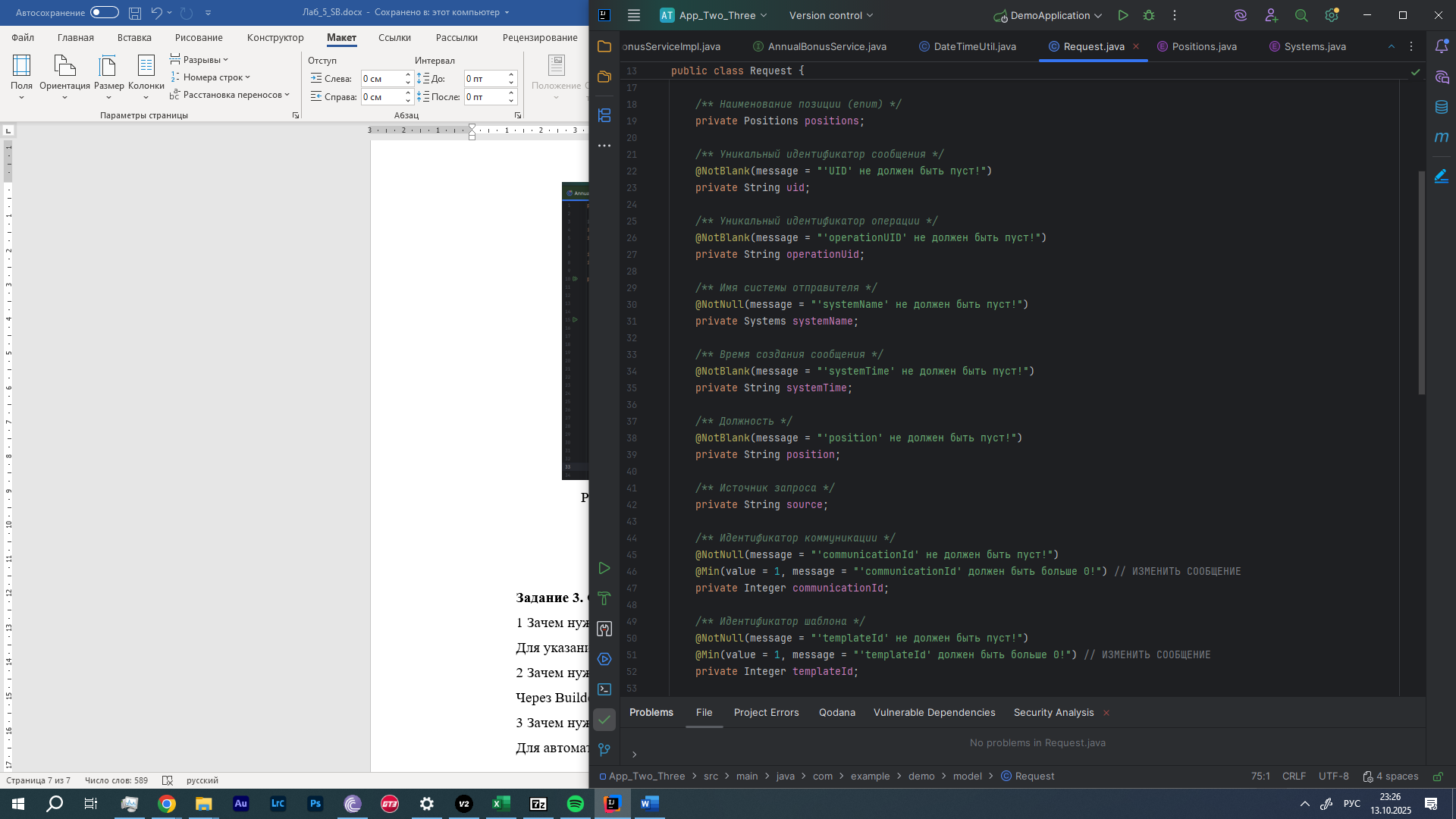


Рисунок 10 – Расставленные в Request комментарии

Теперь нам требуется оптимизировать большой код MyController.java. Сначала убираем пару неиспользуемых библиотек, затем – дублирующиеся логги; и, наконец, используем методы вместо вызова однотипных операций: для вывода логгов – logRequestInfo, для вызова билдера – createInitialResponse, для запроса (и проверки uid==123) – validateRequest, для выброса ошибок – handleException и updateErrorResponse, для модификаций запроса при работе двух сервисов – processModifications. Частичный оптимизированный код представлен на рисунке 11.

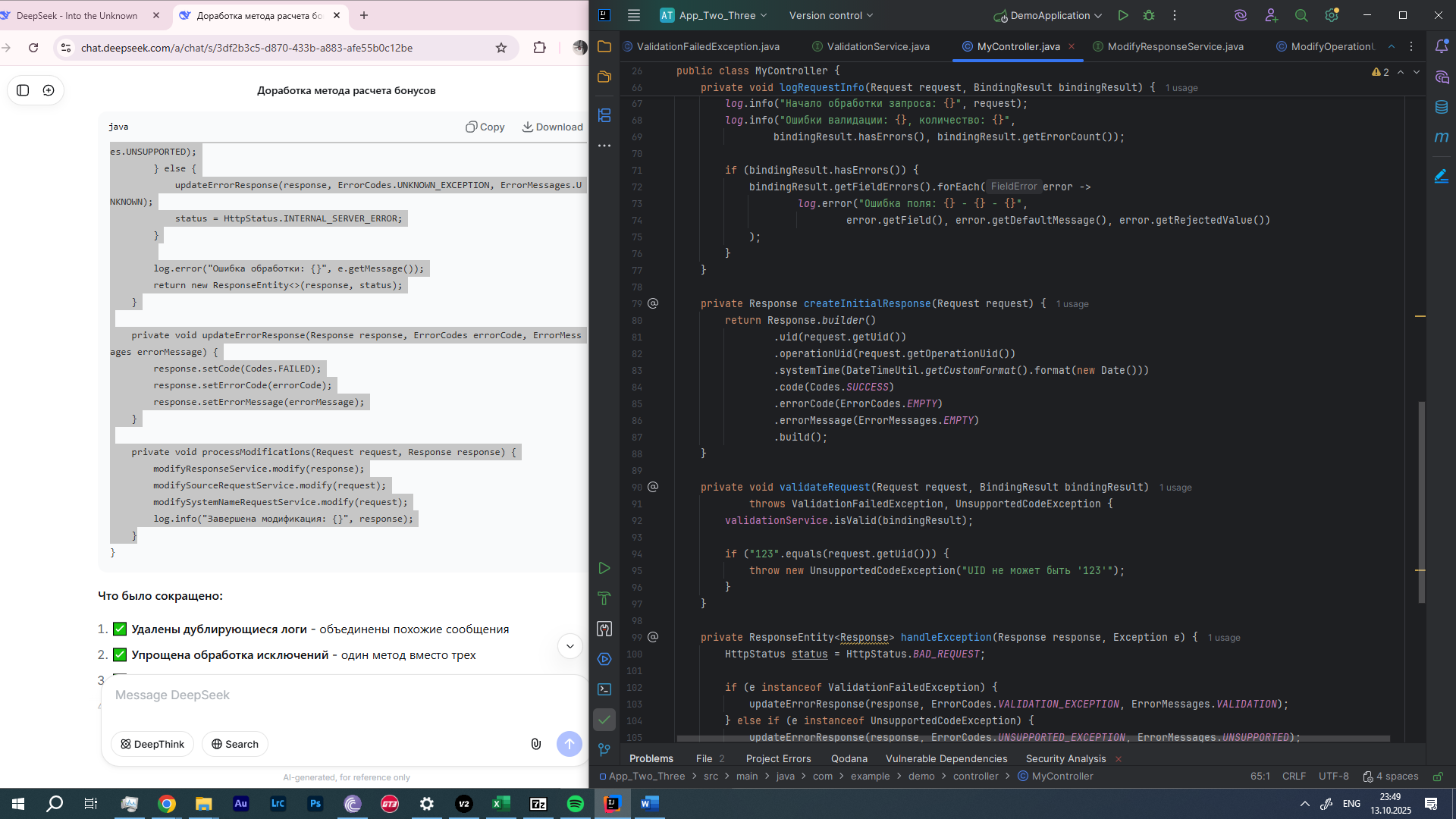


Рисунок 11 – Оптимизированный код MyController

**Задание 3. Ответы на вопросы.**

1 Что такое unit тестирование?

Это тестирование отдельных компонентов и модулей программы вне других частей системы (быстрая проверка одного модуля).

2 Что такое интеграционное тестирование?

Это тестирование взаимодействия между различными компонентами и/или модулями системы (тестирует несколько компонентов сразу, но медленее unit, проверяет интеграцию).

3 Что такое Assert? Какие бывают Assert?

Это проверка утверждения в тесте, проверяет проходит ли тест или нет. Бывают:

а) assertEquals – проверка на равенство;

б) assertTrue – проверка на истинность (true);

в) assertFalse – проверка на истинность (false);

г) assertNull – проверка на null (null);

д) assertNotNull – проверка на null (не null);

е) assertThrows – проверка на исключения.

**Вывод**

В результате лабораторной работы были доработаны простой REST сервис и добавлены unit тесты.