ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ В.Н. КАРАЗІНА

Навчально-науковий інститут комп’ютерних наук та штучного інтелекту

Кафедра інтелектуальних програмних систем і технологій

**ЛАБОРАТОРНА РОБОТА № 1**

на тему: «Spring boot»

дисципліна: «Архітектури і патерни проектування

інформаційних систем»

Виконала: студентка 4 курсу групи КС41

Спеціальності 124 «Комп’ютерні науки»

Манюк В.П

Харків

2025

**Мета:** Ознайомитись з технологією Spring Boot

**Хід роботи**

Для початку роботи було завантажено необхідні пакети для створення проєкту: java -version, brew -v, brew install maven, mvn -version

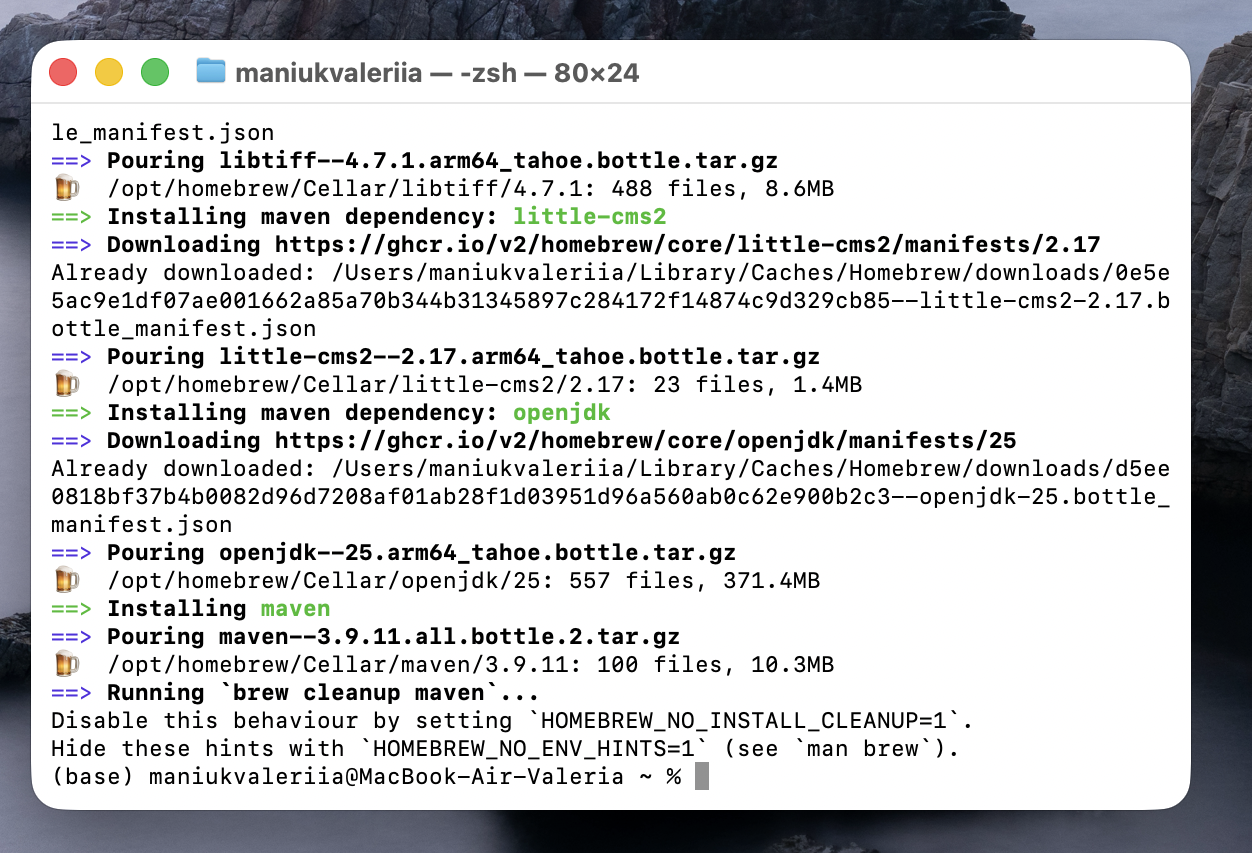


Рисунок 1 - Початок роботи

Створено новий проєкт у середовищі VS Code за допомогою Spring Initializr. Задано назву springbootconsoleapp, групу com.example, артефакт springbootconsoleapp, пакет com.example.springbootconsoleapp. Обрано мову Java, систему збірки Maven та версію JDK 17.

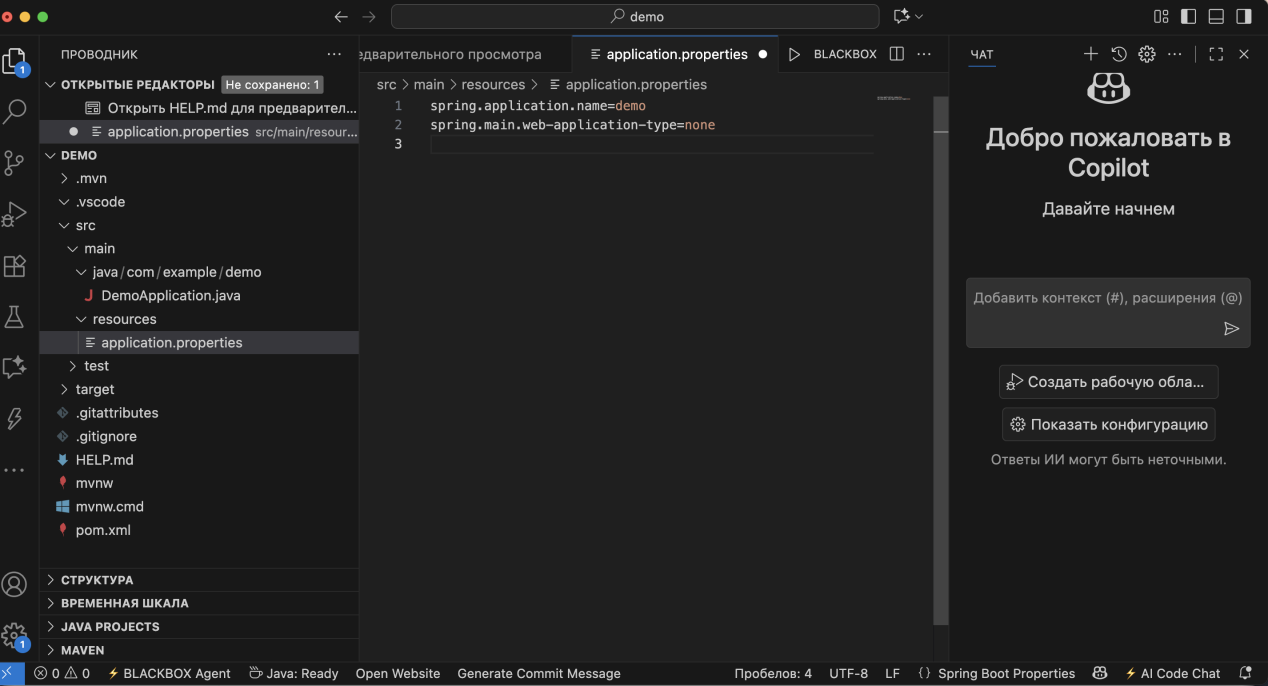


Рисунок 2 – Створення нового проєкту Spring Boot

Після генерації проєкту створено стандартну структуру Maven: каталоги src/main/java, src/main/resources та src/test/java. IntelliJ згенерувала початковий клас Lab1Application та файл конфігурації application.properties.

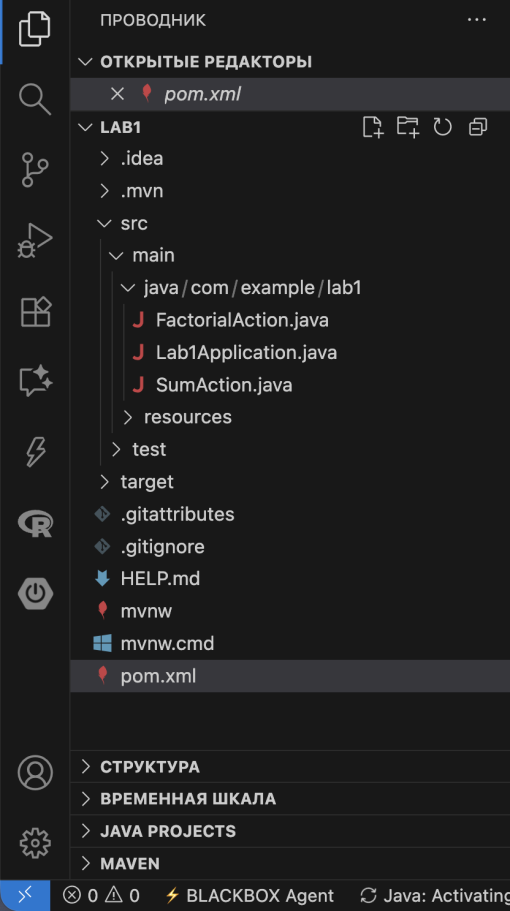


Рисунок 3 – Структура створеного проєкту

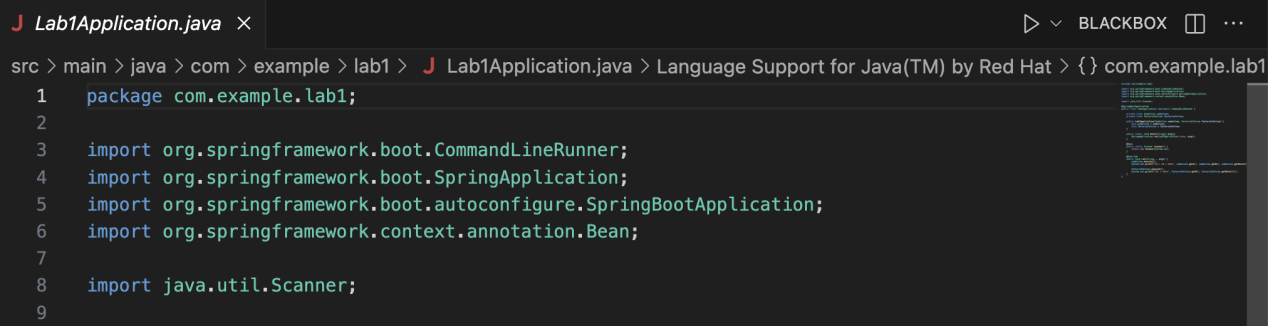


Рисунок 4 – Початковий клас Lab1Application

Автоматично згенерований клас Lab1Application містить метод main() із запуском Spring Boot через SpringApplication.run().

До цього класу додано реалізацію інтерфейсу CommandLineRunner, що дозволяє виконувати власний код після старту застосунку у методі run().

У цьому методі здійснюється ін’єкція залежності — об’єкт класу SumAction передається через конструктор (Dependency Injection). Далі викликається метод execute(), який зчитує вхідні дані користувача, обчислює суму введених чисел і виводить результат у консоль.

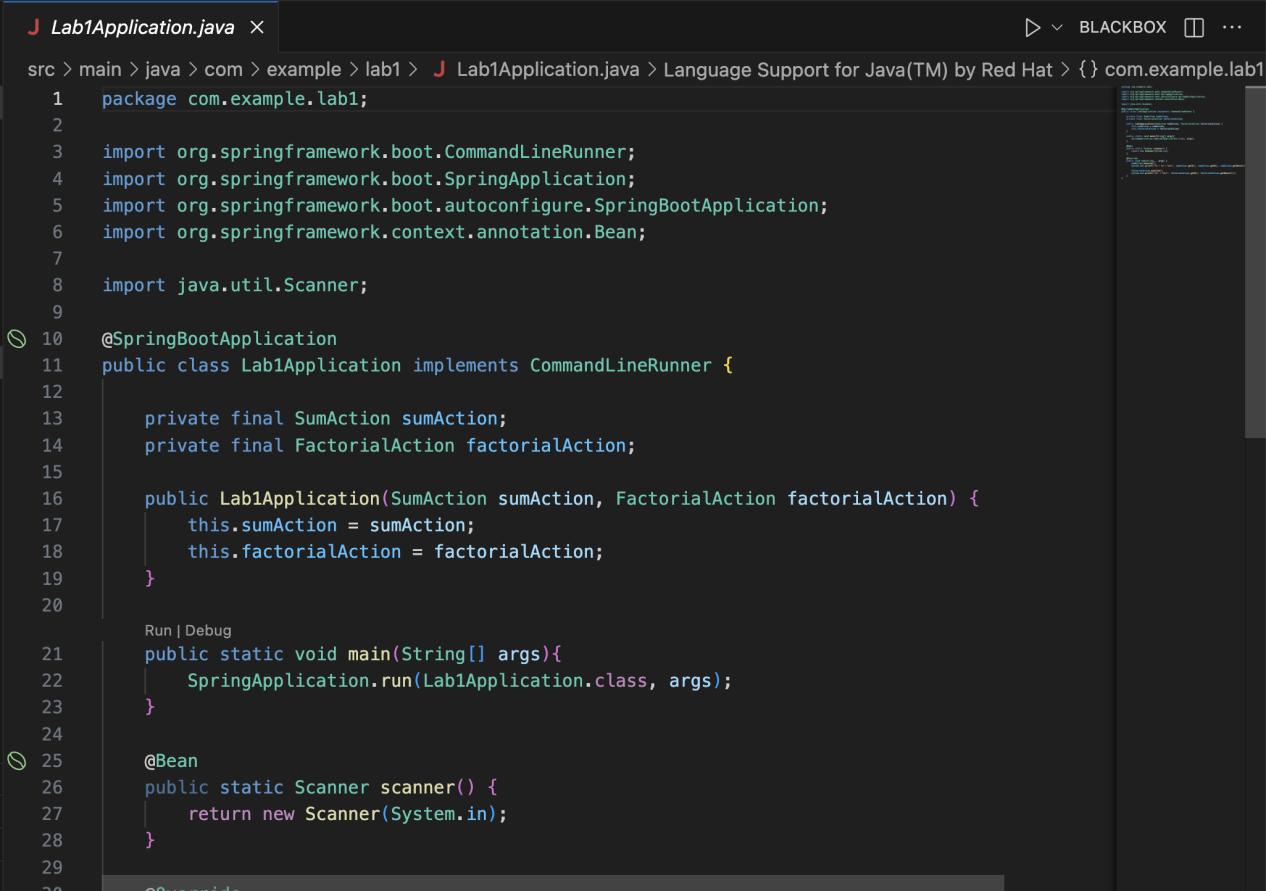


Рисунок 5 – Модифікований клас Lab1Application

У конфігураційний файл додано параметр:

spring.main.web-application-type=NONE

Це вимикає веб-режим і дозволяє запускати застосунок як консольний.

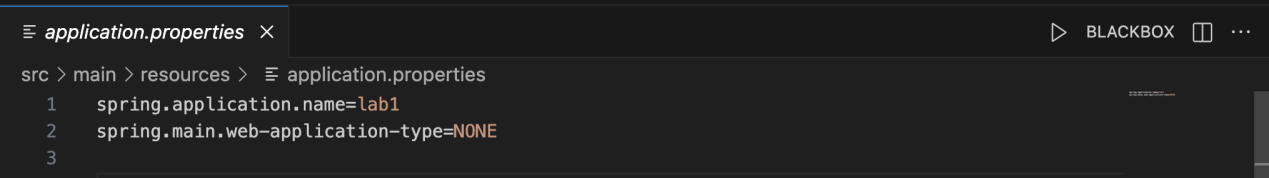


Рисунок 6 – Файл application.properties

Було створено клас SumAction, який містить поля a, b та result. Залежність Scanner впроваджено через ін’єкцію залежностей (Dependency Injection) — об’єкт Scanner передається у конструктор класу. Основна логіка розміщена в методі execute(), який зчитує два цілі числа з клавіатури, обчислює їх суму та зберігає результат у полі result. Клас має гетери для доступу до значень змінних.

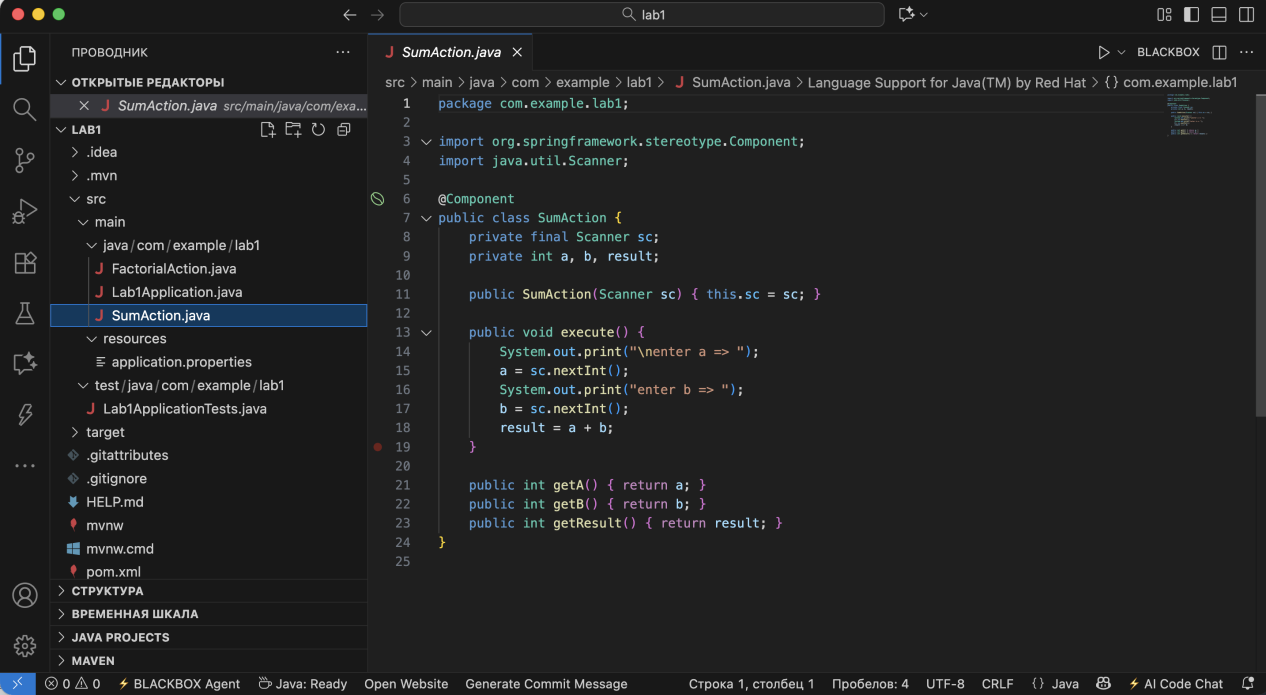


Рисунок 7 – Клас SumAction

Після запуску програми у консолі виводиться банер Spring Boot та службові повідомлення. Далі користувач вводить два числа, після чого програма обчислює їх суму та завершує роботу.

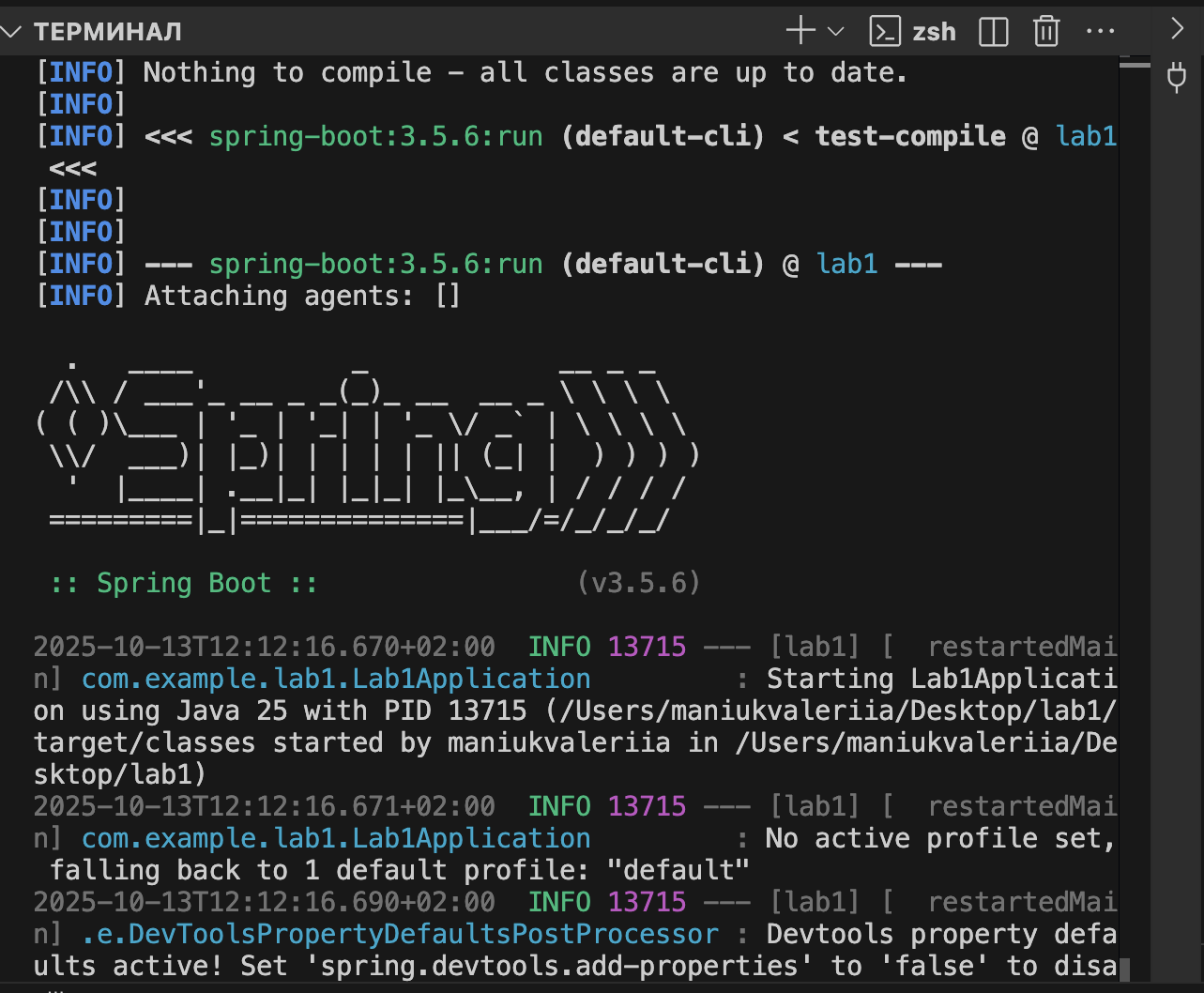


Рисунок 8 - Результат запуску програми

У класі Lab1Application додано створення ще одного об’єкта — FactorialAction. Тепер програма виконує не тільки додавання, а й обчислення факторіала.

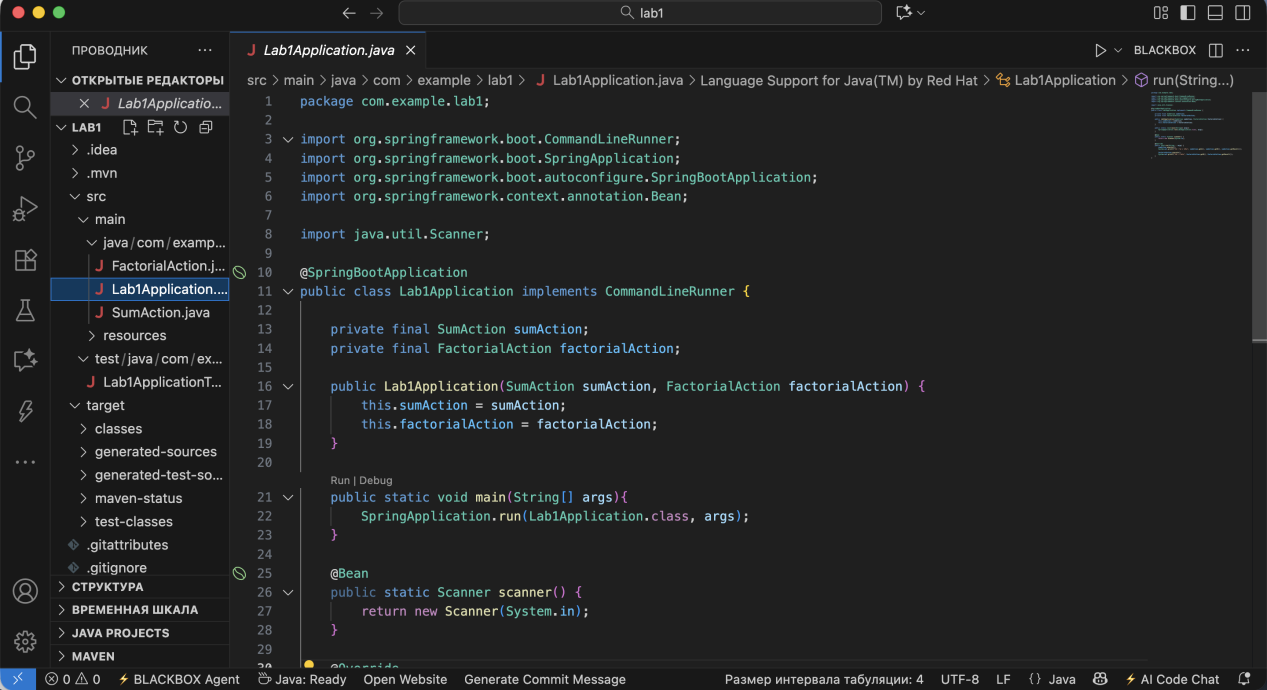


Рисунок 9 – Розширений клас Lab1Application

Клас FactorialAction, який реалізує обчислення факторіала числа. У класі використано ін’єкцію залежностей (Dependency Injection) — об’єкт Scanner передається через конструктор. Метод execute() зчитує з клавіатури значення n, обчислює факторіал у циклі та зберігає результат у полі result. Клас має гетери getN() та getResult() для доступу до відповідних полів.

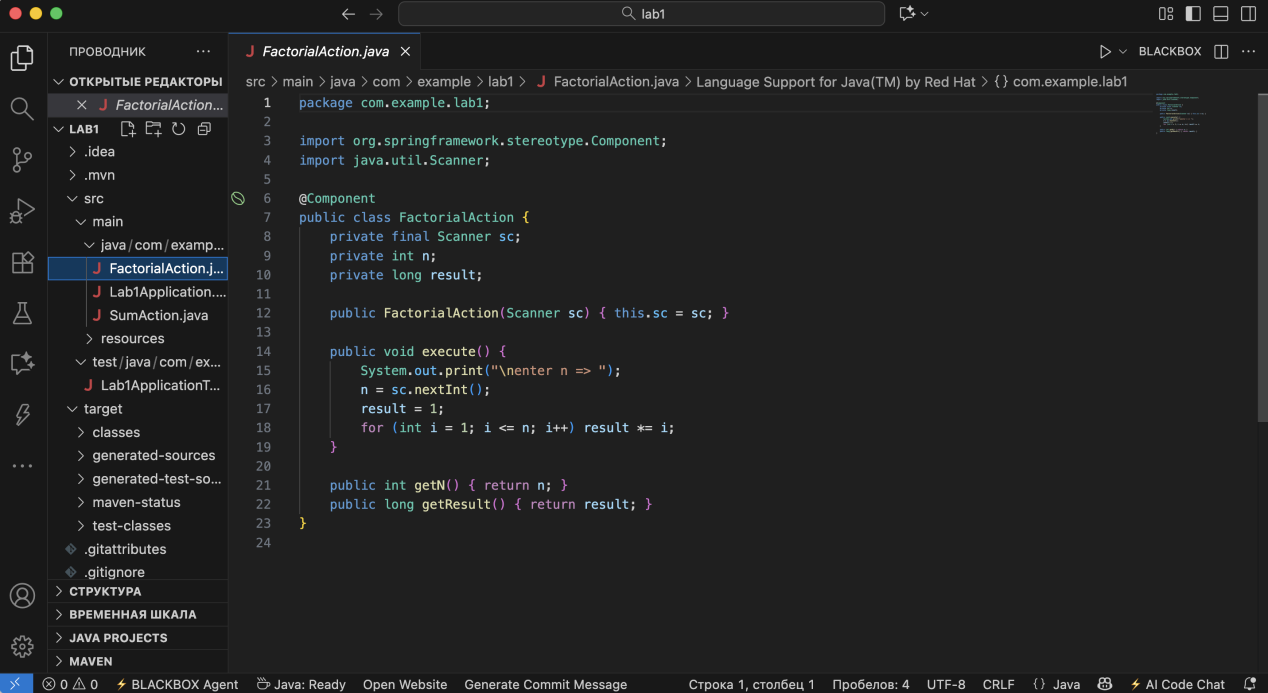


Рисунок 10 — Клас FactorialAction із реалізацією через ін’єкцію залежностей

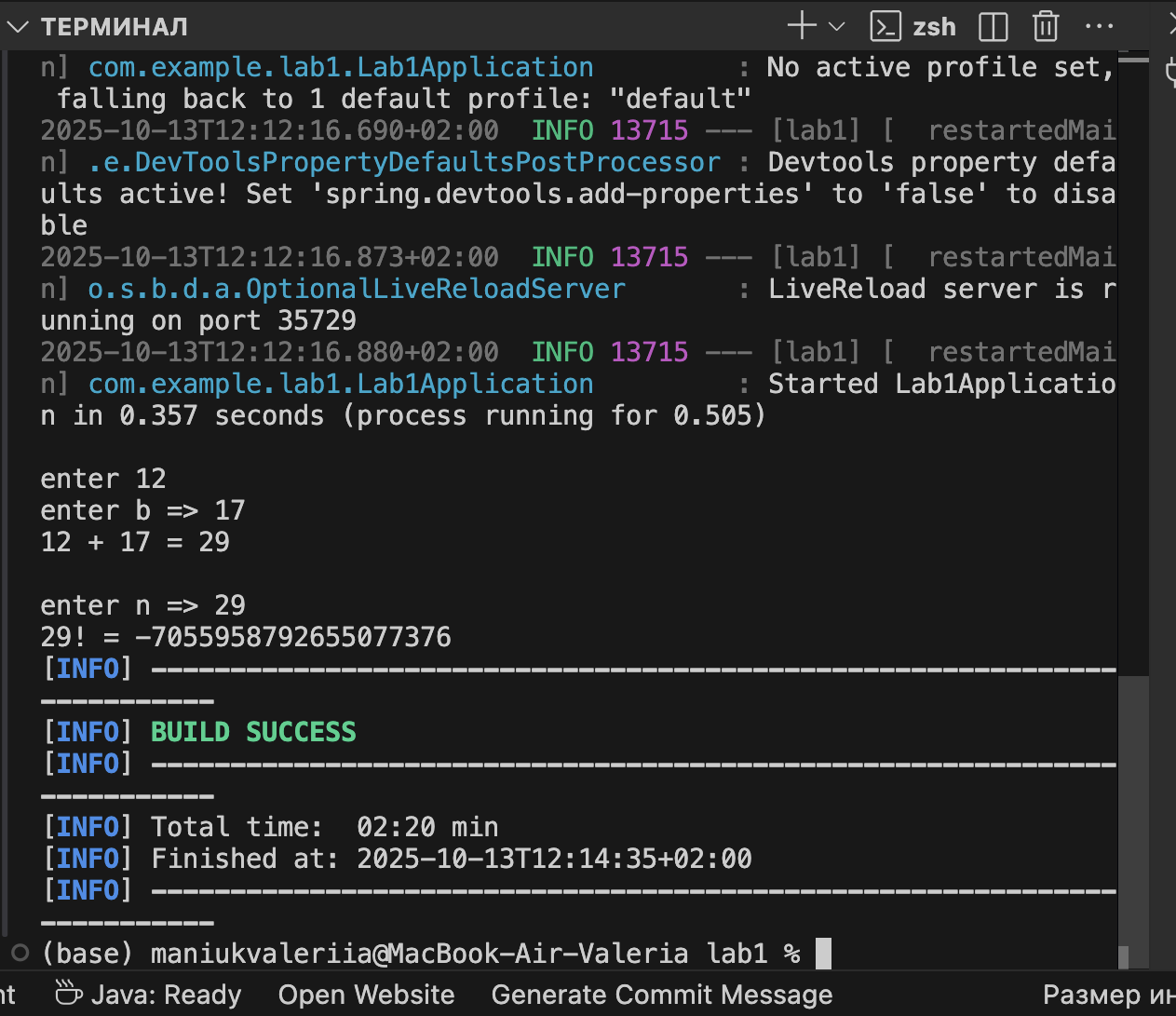


Рисунок 11 - Результат запуску програми

У консолі відображено приклад роботи програми: користувач вводить числа, після чого виводиться результат додавання та факторіал введеного числа.

**Контрольні запитання**

1. Які переваги використання Spring Boot?

Spring Boot спрощує створення застосунків завдяки автоматичній конфігурації, вбудованому вебсерверу, готовим залежностям (starter’ам) і мінімальній кількості ручних налаштувань.

1. Які ключові компоненти Spring Boot?

Основні компоненти: Spring Boot Starters (набір готових залежностей), Spring Boot AutoConfiguration (автоматичне налаштування), Spring Boot CLI (консольний інструмент) та Actuator (моніторинг і керування).

1. Як працює Spring Boot?

Spring Boot автоматично підключає потрібні залежності й виконує конфігурацію на основі вмісту проєкту. Це дозволяє швидко запускати застосунок без тривалого налаштування.

1. Як можна увімкнути функції Spring Boot?

Функції активуються через залежності у pom.xml або build.gradle, використання анотацій (наприклад, @SpringBootApplication, @EnableAutoConfiguration) і налаштувань у файлі application.properties чи application.yml.

1. Як запускається програма для завантаження Spring?

Програма запускається через метод main(), у якому викликається SpringApplication.run(...). Це створює і завантажує Spring Application Context.

6. Для чого потрібна анотація @SpringBootApplication?

Ця анотація поєднує три інші (@Configuration, @EnableAutoConfiguration, @ComponentScan) і вмикає автоматичне налаштування, конфігурацію та сканування компонентів у проєкті.

**Висновки**

Під час виконання лабораторної роботи було створено консольний застосунок на основі **Spring Boot** для обчислення факторіала числа. Проєкт згенеровано за допомогою **Spring Initializr**, обрано систему збірки **Maven**, мову **Java** та додано необхідні залежності. У процесі реалізації розроблено клас **FactorialService**, який виконує обчислення, та головний клас, що забезпечує запуск і відображення результатів у консолі. Результати виконання підтвердили правильну роботу застосунку. У ході роботи було закріплено навички створення консольних застосунків у Spring Boot і налаштування їх параметрів для консольного режиму.