Національний технічний університет України

«КПІ ім. Ігоря Сікорського»

Факультет Інформатики та Обчислювальної Техніки

Кафедра Інформаційних систем та технологій

**Звіт до лабораторної роботи №1**

з дисципліни «Розроблення застосунків з використанням Spring Framework»  
на тему «Перше знайомство з Spring Framework»

Виконали:

Студенти групи ІС-34

Майко Денис Ростиславович

Кудрявцев Артем Максимович

Перевірив:

Асистент кафедри ІСТ, ФІОТ

Нікітін Валерій Андрійович

Київ – 2025

**МЕТА**: Ознайомитись з принципами функціонування, створення, навчання та використання моделей машинного навчання.

**ПОСТАНОВКА ЗАВДАННЯ:**

1. Встановіть JDK. При цьому зверніть увагу:

- версія JDK - не нижче ніж JDK8 (бажано – останню LTS);

- не переплутайте JDK (Java Development Kit) з JRE (Java Runtime Environment);

- пропишіть у системі змінні PATH та JAVA\_HOME.

2. Встановіть IDE:

- Spring Tool Suit (Free, Open Source)

https://spring.io/tools

- IntelliJ IDEA Ultimate Edition (як студенти КПІ ви можете отримати безкоштовну ліцензію)

https://www.jetbrains.com/student/

3. Створіть і запустіть консольний застосунок HelloWorld

3.1. За допомогою Spring Starter створіть проект з наступними параметрами

\* Версії Java або інших елементів конфігурації можуть відрізнятися від тих, що вказані на рисунках.

\*\* При першому використанні процес завантаження залежностей у локальний репозиторій може зайняти деякий час в залежності від швидкості Вашого підключення до мережі. Наступні проєкти будуть створюватись значно швидше.

3.2. Додайте до декларації основного класу інтерфейс CommandLineRunner та відповідний метод run(String... agrs).

3.3. Запустіть проект.

Зверніть увагу на порядок виводу рядочків у консолі.

4. Створіть і запустіть веб-застосунок HelloWorld

4.1. Оберіть в меню File -> New -> Spring Starter Project. Задайте наступні параметри:

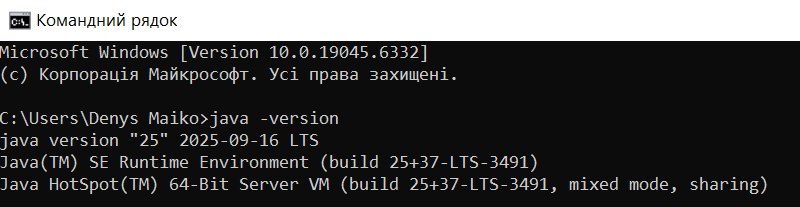
4.2. Додайте контролер.

4.3. Запустіть проект, відкрийте браузер та перейдіть за посиланням http://localhost:8080/

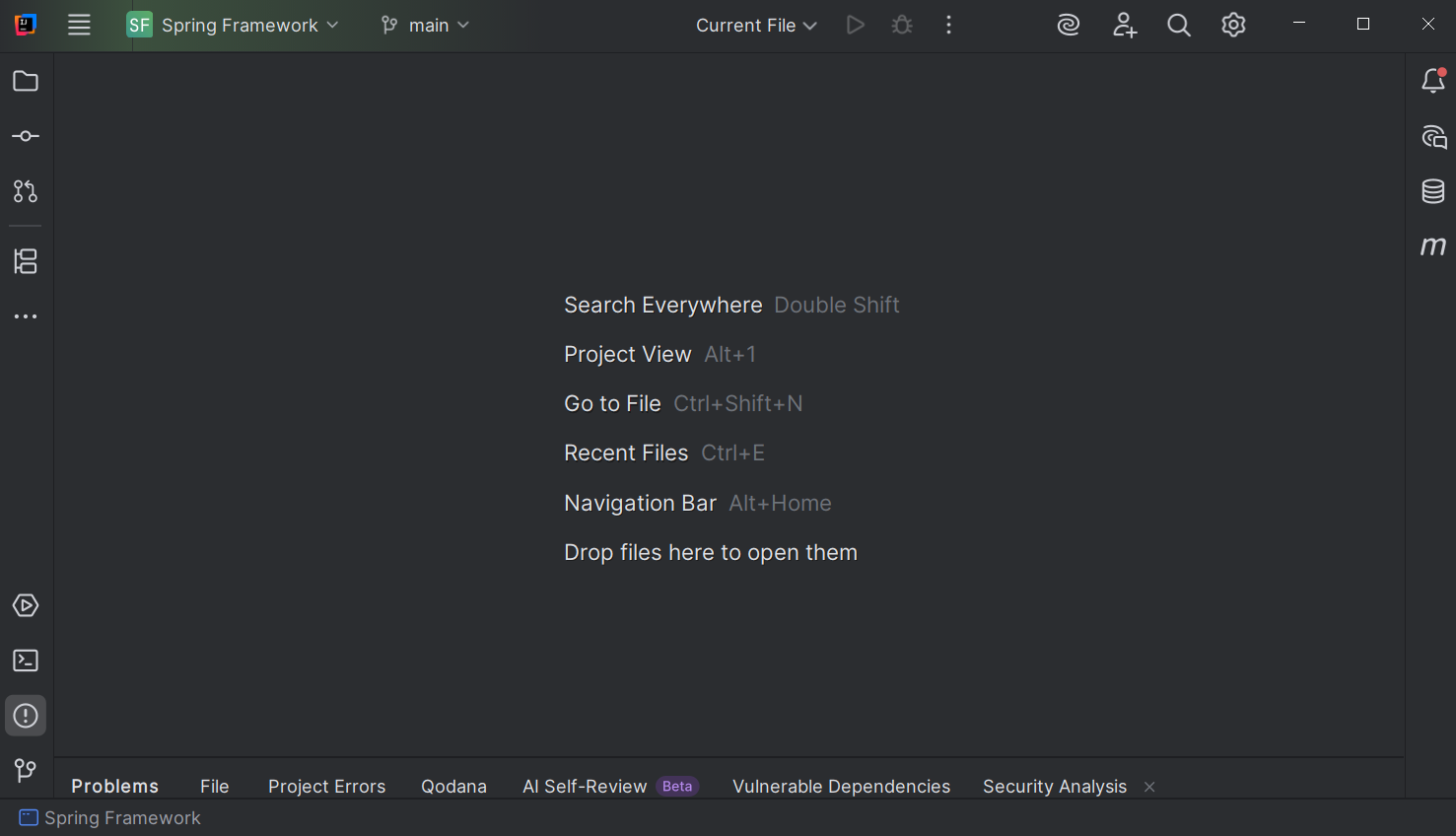
**РОЗВ’ЯЗАННЯ:**

Посилання на репозиторій GitHub: <https://github.com/DenysMaiko/SpringTeam>

1. Встановимо JDK та перевіримо наявність через термінал CMD.

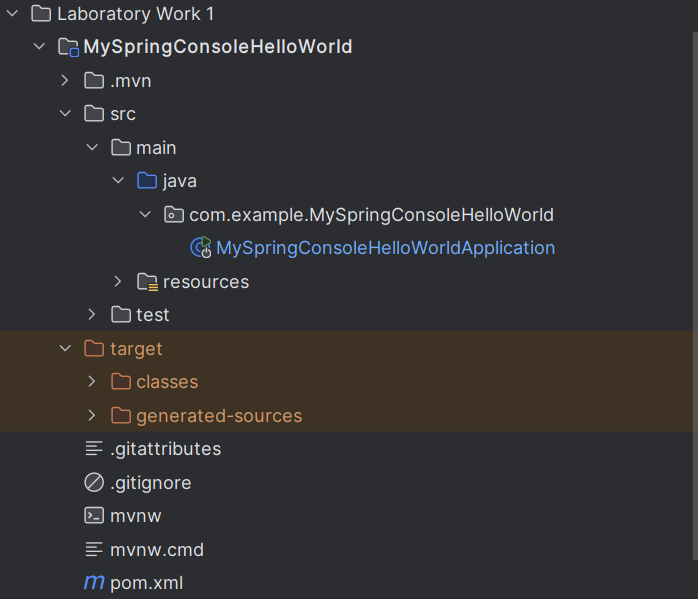


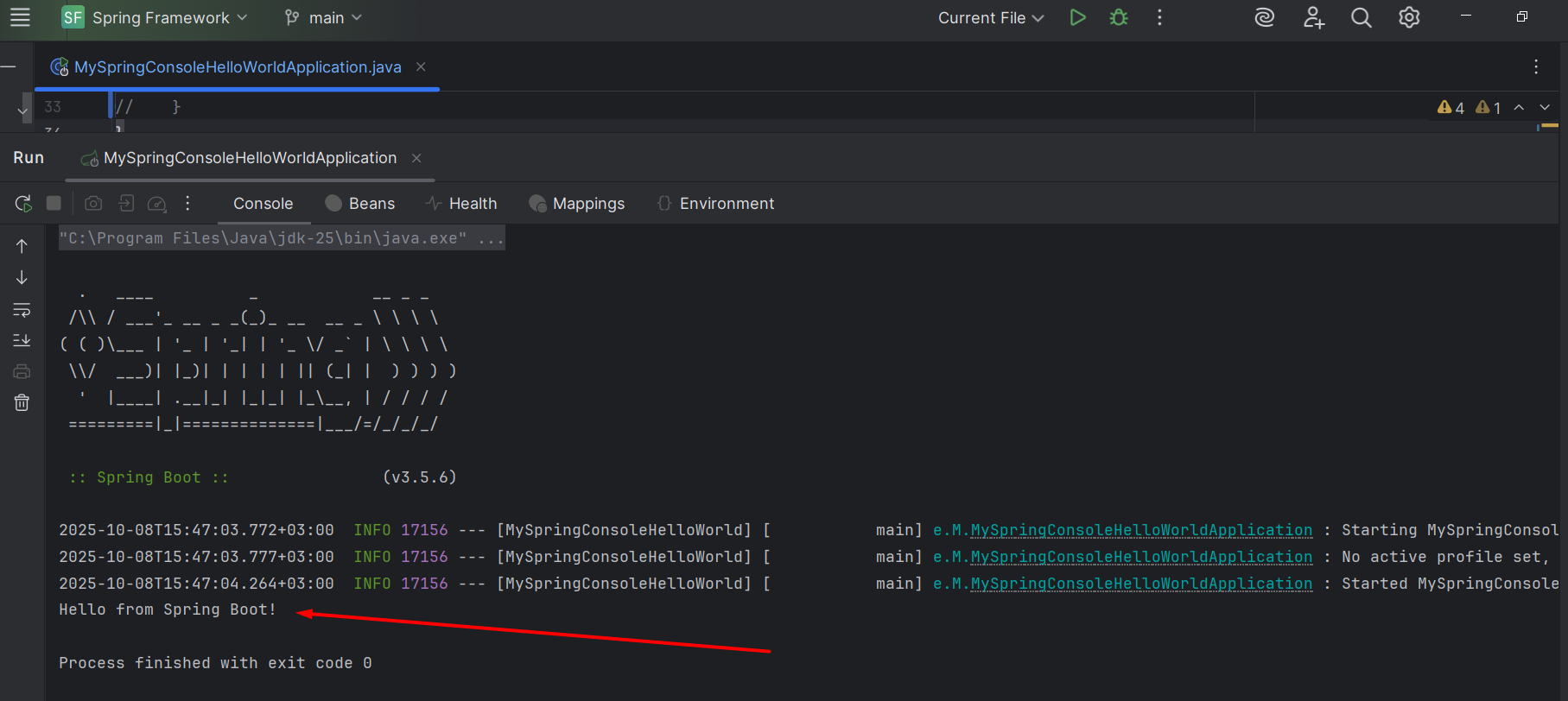
2. Встановимо IDE IntelliJ IDEA Ultimate Edition.



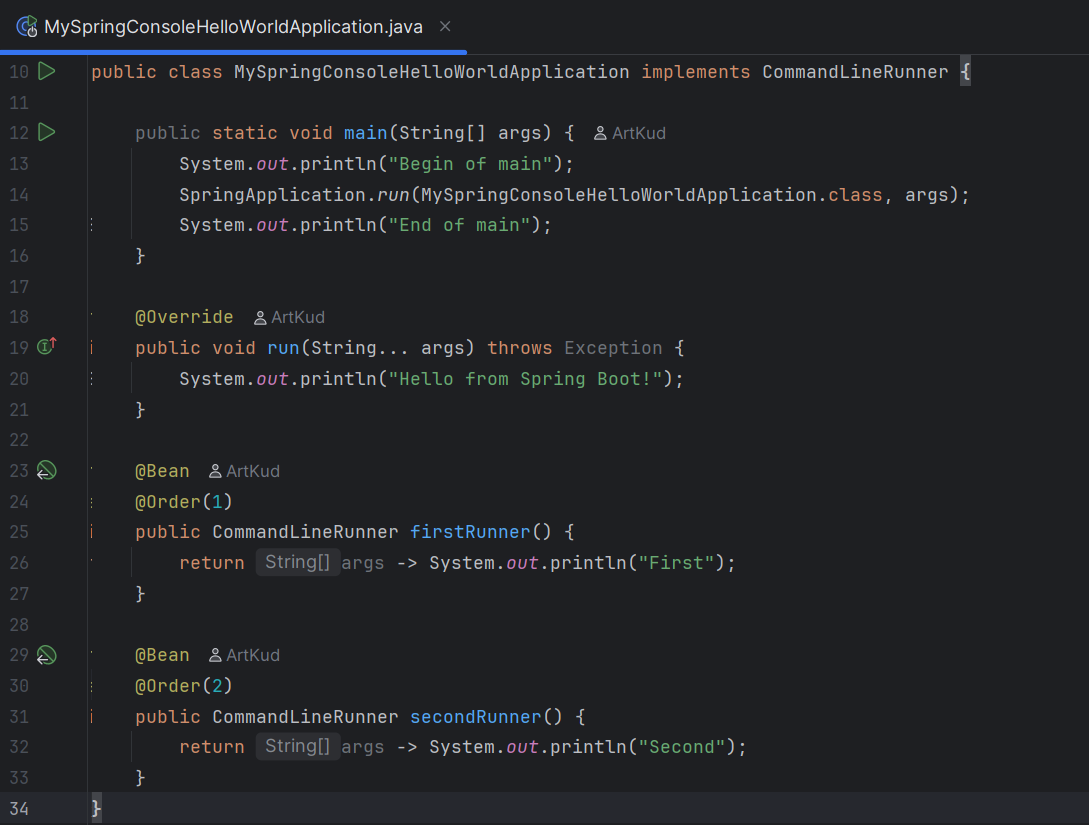
3. Створити та запустити консольний застосунок HelloWorld.

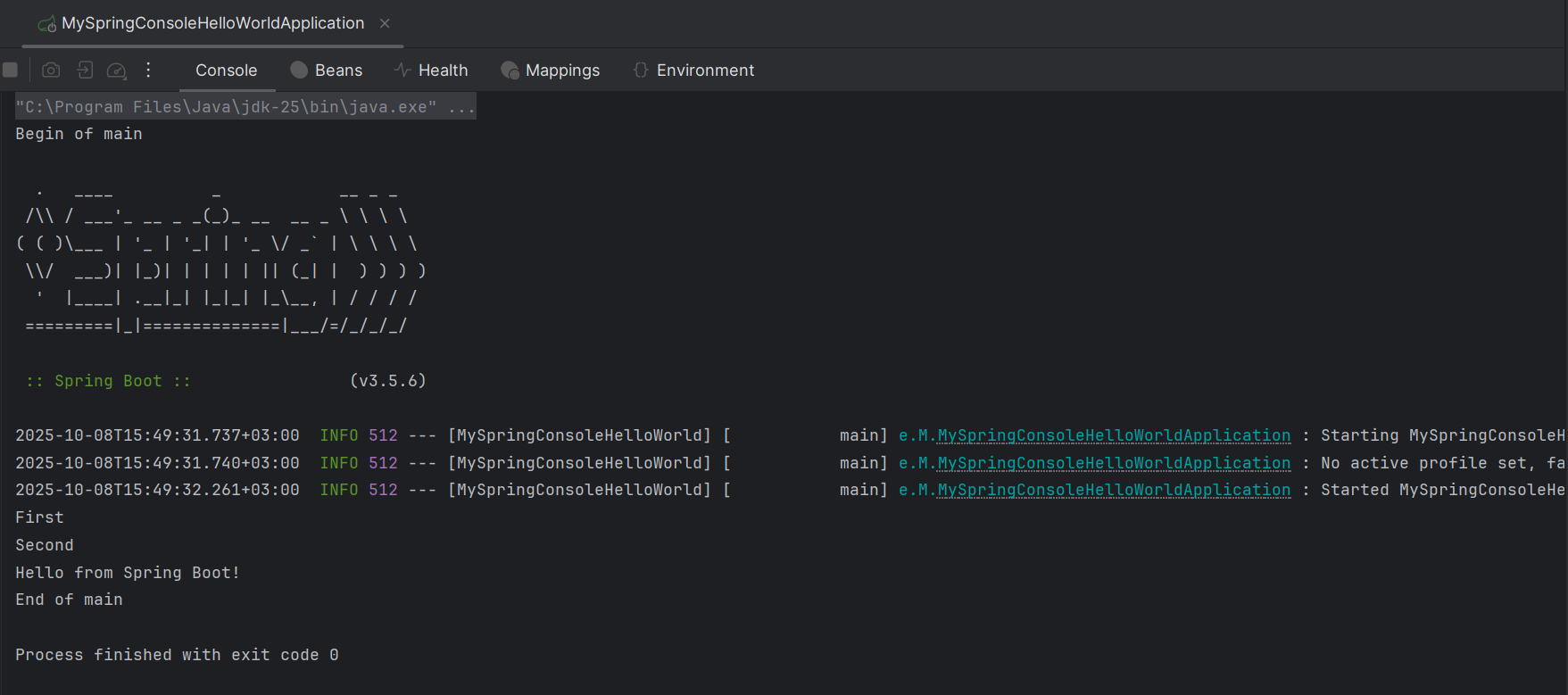
Структура проекту:





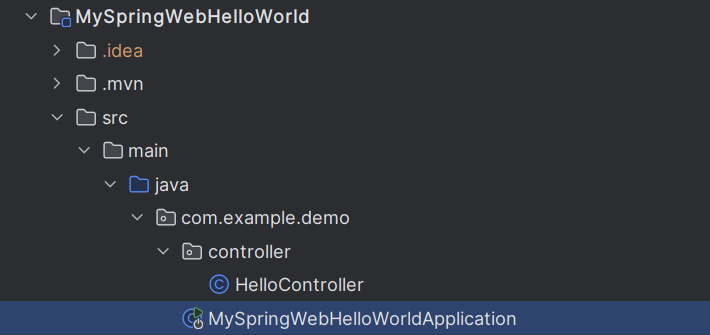
До контрольного питання 7

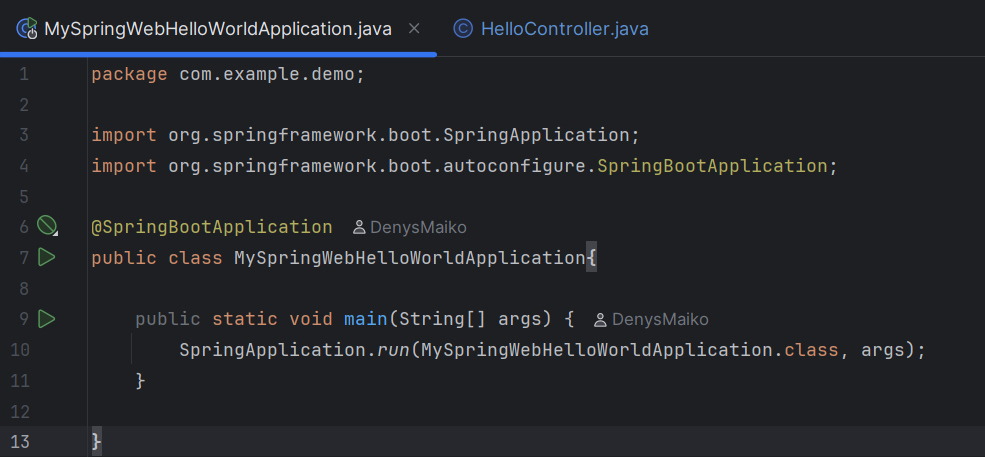


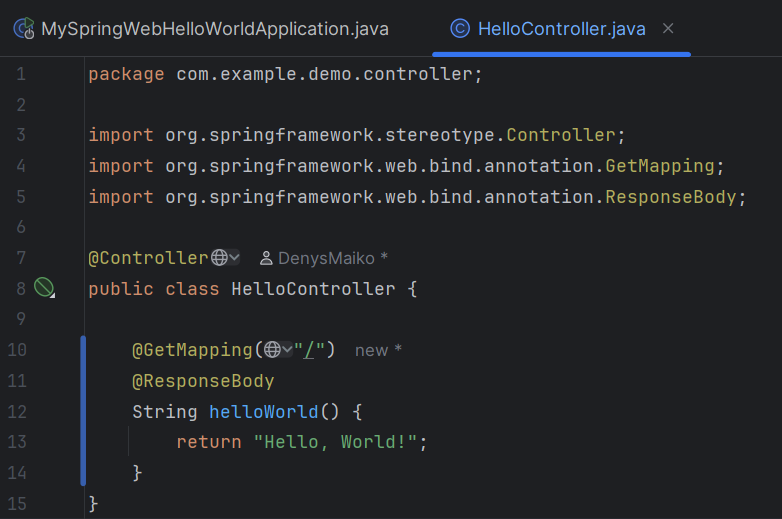


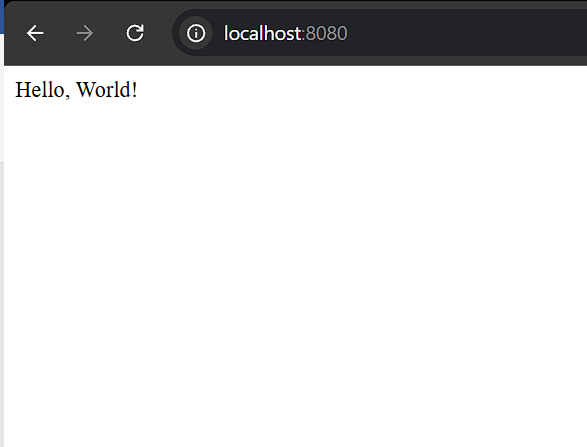
4. Створимо і запустимо веб-застосунок HelloWorld.

Структура проекту:



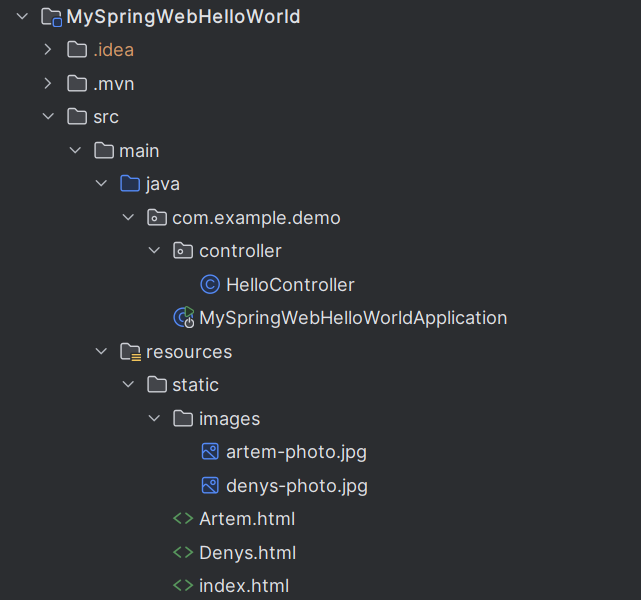


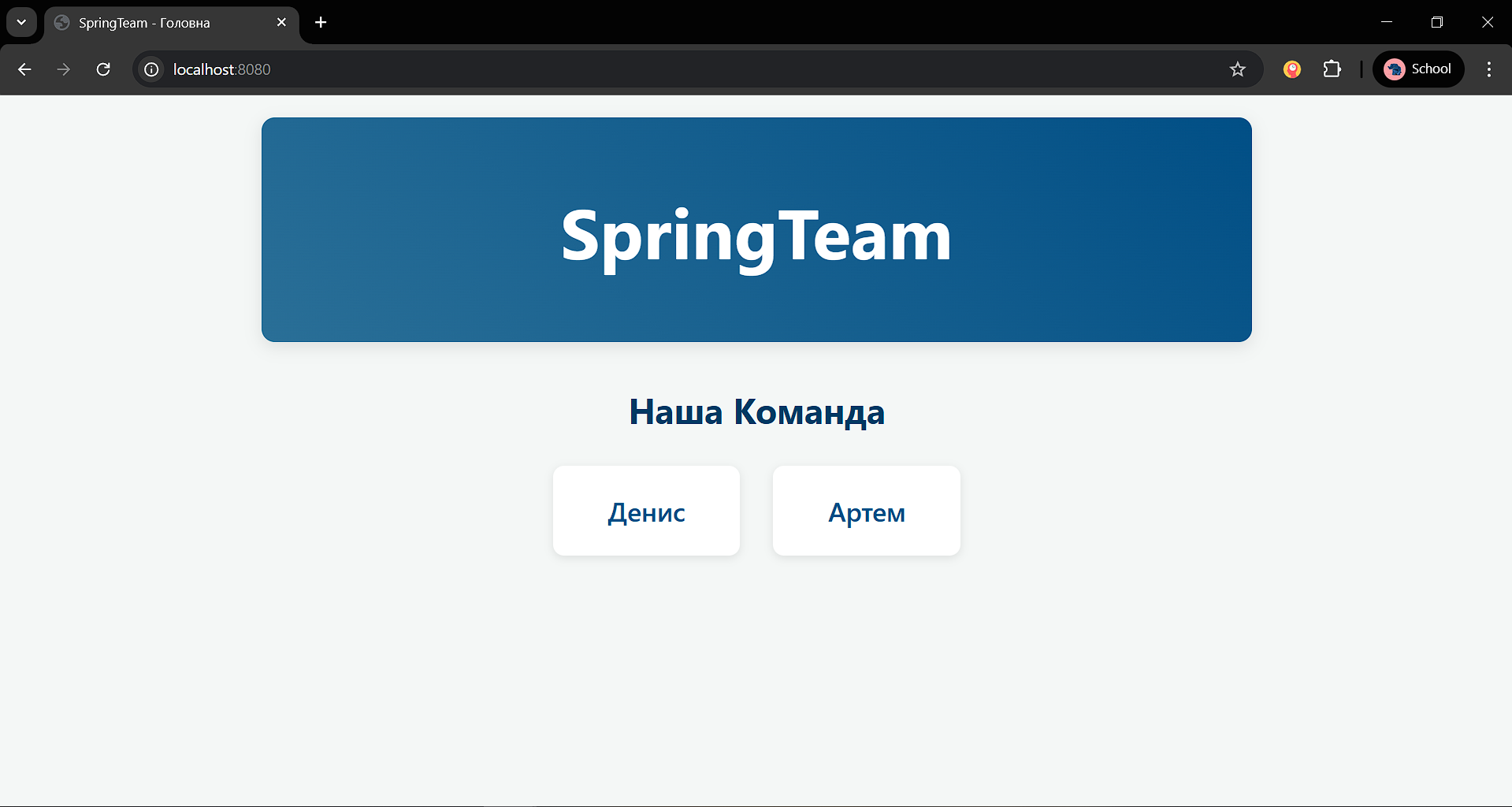


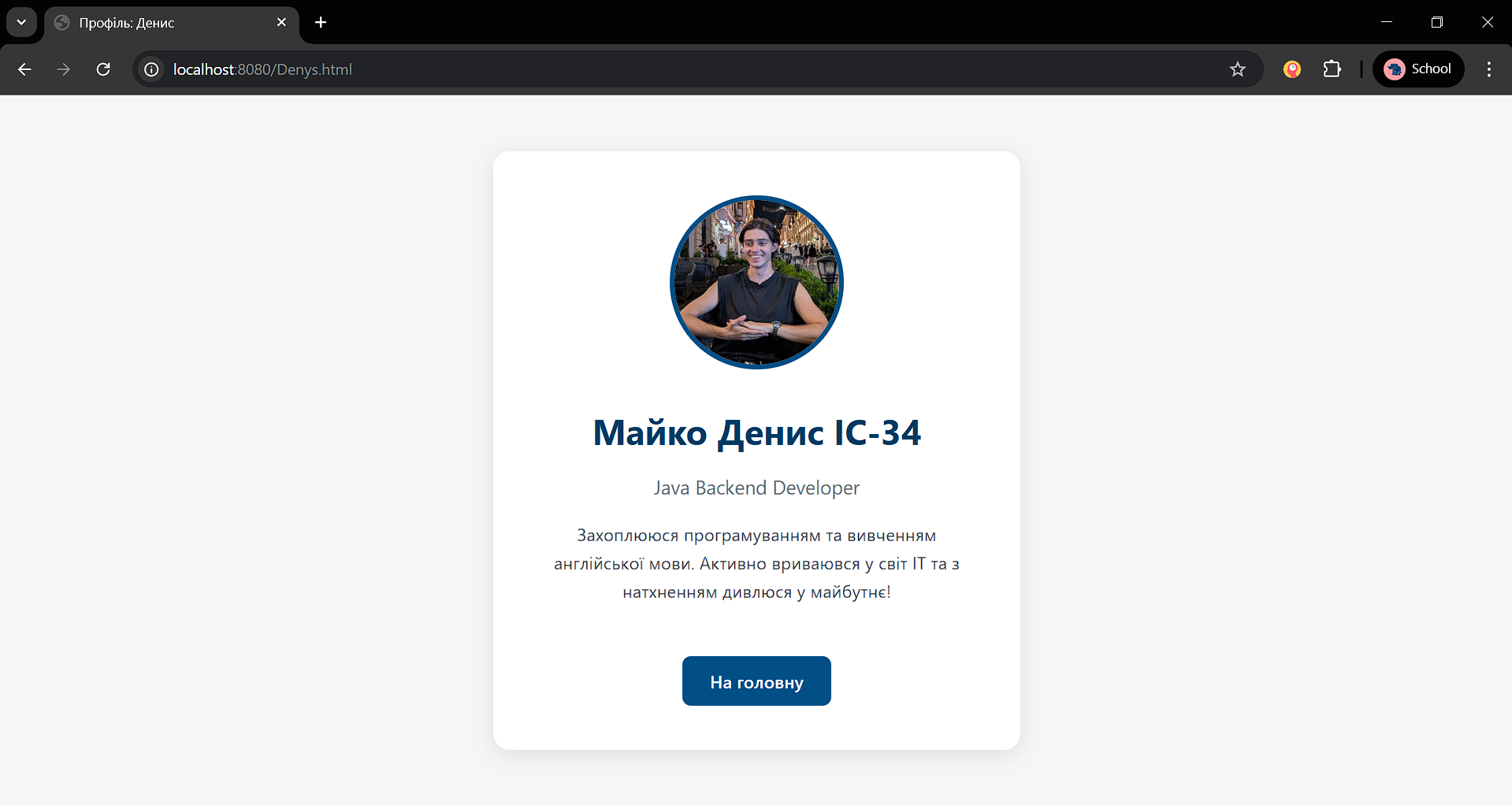


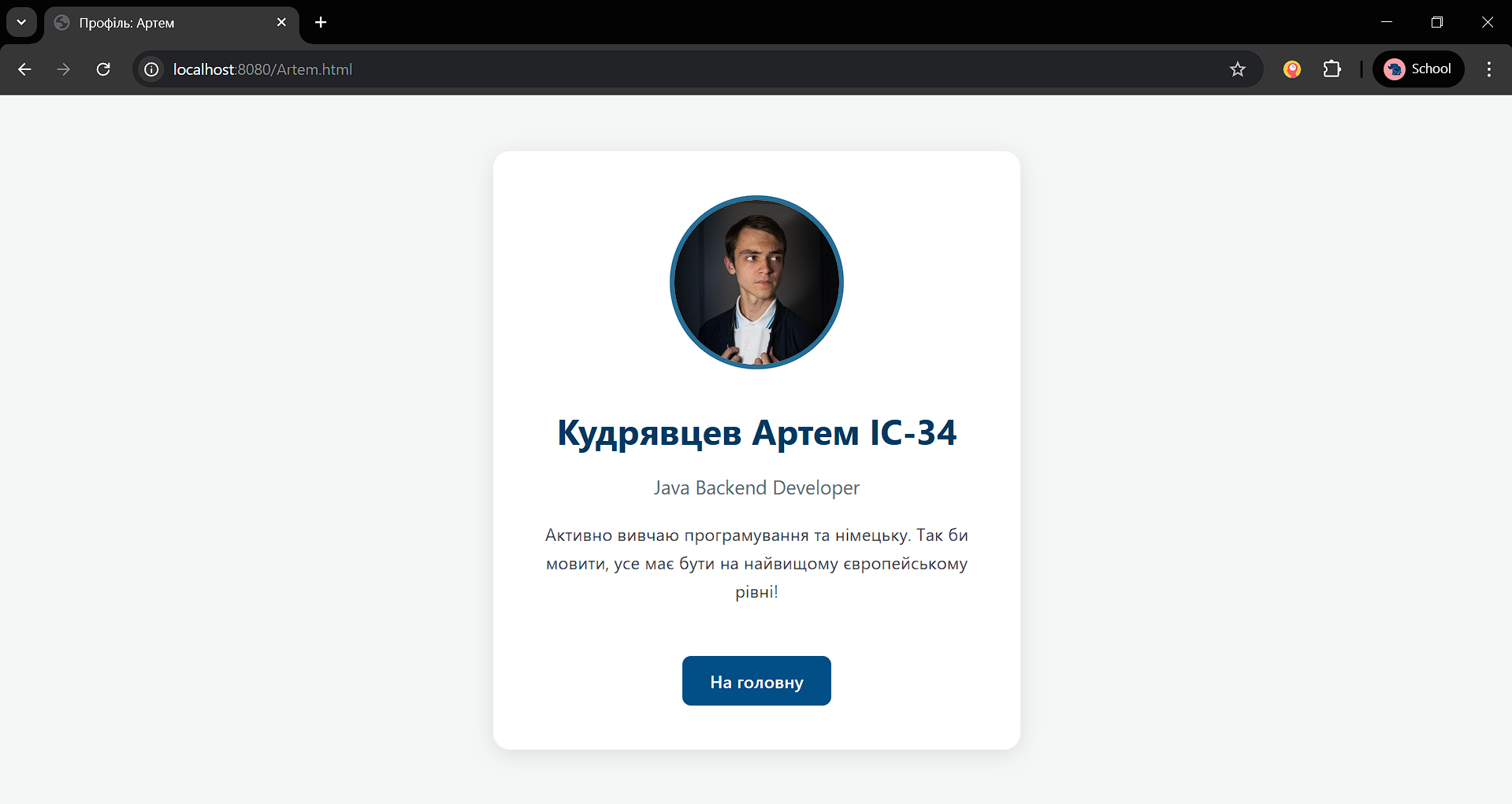
До контрольного питання 8

Структура проекту:









**ВИСНОВОК:**

В ході даної лабораторної роботи ми попрактикувалися у роботі та використанні Spring Framework та його розширення Spring Boot. Створили пару проектів з різними початковими параметрами та залежностями на основі системи управління проектами та збірками Maven. Попрактикувалися у виведенні тексту у консоль (до та після виведення інформації про логи), на сторінці серверу та через статичну сторінку html. Розглянули чим відрізняється фреймворк від бібліотеки, яка працює CommandLineRunner та ApplicationRunner, анотація @SpringBootApplication тощо. Освоїли практичні навички роботи у середовищі IntelliJ IDEA Ultimate Edition, завантажили усі необхідні файли у сервіс системи контролю версій GitHub.

**Контрольні питання:**

**1. В чому полягає різниця між бібліотекою та фреймворком?**

Основна відмінність між бібліотекою та фреймворком полягає у концепції інверсії контролю (Inversion of Control, IoC).

* Бібліотека (Library) є набором функцій та класів, які розробник використовує у своєму коді. У цьому випадку потік виконання програми контролюється кодом розробника, який у потрібні моменти викликає методи з бібліотеки.
* Фреймворк (Framework), навпаки, сам диктує структуру додатку та керує потоком виконання. Він викликає код, написаний розробником, у відповідні моменти свого життєвого циклу. Таким чином, фреймворк надає каркас, який розробник наповнює власною бізнес-логікою.

**2. В чому полягає різниця між Spring Framework та Spring Boot?**

Spring Framework — це фундаментальний фреймворк, що надає комплексні модулі для розробки Java-додатків, зокрема для впровадження залежностей (Dependency Injection), веб-розробки (Spring MVC) та доступу до даних. Однак він вимагає значної кількості ручної конфігурації для поєднання всіх компонентів.

Spring Boot — це розширення (надбудова) над Spring Framework, створене для максимального спрощення розробки. Його ключові переваги:

* Автоконфігурація: Автоматично налаштовує додаток на основі залежностей, що значно зменшує кількість шаблонного коду.
* Стартові залежності (Starters): Надає готові набори залежностей для типових завдань (веб, робота з базами даних), спрощуючи управління проєктом.
* Вбудований сервер: Дозволяє запускати веб-додатки як самостійні .jar файли без необхідності розгортання на зовнішньому веб-сервері.

Отже, Spring Boot не замінює Spring Framework, а прискорює та спрощує роботу з ним.

**3. Для чого в структурі проекту потрібен файл pom.xml?**

Файл pom.xml (Project Object Model) є основним конфігураційним файлом для системи автоматизації збірки Apache Maven. Він виконує три головні функції:

* Управління залежностями: У файлі перераховуються всі зовнішні бібліотеки та фреймворки, від яких залежить проєкт. Maven автоматично завантажує та підключає їх.
* Конфігурація збірки: Файл містить інструкції щодо процесу збірки проєкту: компіляції коду, запуску тестів, пакування у фінальний артефакт (наприклад, .jar або .war файл).
* Зберігання метаданих: Включає основну інформацію про проєкт, таку як його назва, версія, опис та інша проектна документація.

**4. Для чого потрібна анотація @SpringBootApplication? Що буде, якщо її прибрати?**

Анотація @SpringBootApplication є центральною для будь-якого Spring Boot додатку. Це зручна мета-анотація, яка поєднує в собі три інші важливі анотації:

* @Configuration: Позначає клас як джерело визначень бінів (компонентів) для контексту додатку.
* @EnableAutoConfiguration: Вмикає механізм автоматичної конфігурації Spring Boot, який налаштовує додаток на основі залежностей у classpath.
* @ComponentScan: Активує сканування компонентів (наприклад, @Controller, @Service) у поточному пакеті та його підпакетах.

Якщо прибрати анотацію @SpringBootApplication, додаток втратить усі характеристики проєкту Spring Boot. Механізм автоконфігурації не запуститься, компоненти не будуть знайдені та зареєстровані в контексті, і, як наслідок, додаток не зможе коректно стартувати.

**5. Для чого потрібен інтерфейс CommandLineRunner?**

CommandLineRunner — це функціональний інтерфейс, який надає можливість виконати блок коду на етапі запуску додатку. Метод run(), визначений у цьому інтерфейсі, викликається автоматично після того, як контекст додатку Spring повністю завантажений, але перед тим, як додаток почне приймати зовнішні запити.

Це використовується для ініціалізаційних завдань, таких як:

* Заповнення бази даних початковими (seed) даними.
* Ініціалізація кешу.
* Виконання одноразових налаштувань або міграцій.

**6. Чим інтерфейс CommandLineRunner відрізняється від ApplicationRunner?**

Обидва інтерфейси, CommandLineRunner та ApplicationRunner, служать одній меті — виконання логіки при старті додатку. Ключова і єдина відмінність між ними полягає у способі доступу до аргументів командного рядка.

* CommandLineRunner: Його метод run(String... args) приймає аргументи командного рядка як простий масив рядків (String[]).
* ApplicationRunner: Його метод run(ApplicationArguments args) приймає аргументи у вигляді об'єкта ApplicationArguments. Цей інтерфейс надає більш структурований доступ до аргументів, дозволяючи зручно розрізняти іменовані опції (наприклад, --port=8080) та звичайні аргументи.

Таким чином, ApplicationRunner пропонує більш гнучкий механізм для парсингу аргументів запуску.