

Spring Framework

Lập trình viên Backend - Cybersoft Education

<https://cybersoft.edu.vn/>

Ứng dụng được xây
dựng như thế nào?



Yêu cầu của khách hàng !

- Một ứng dụng luôn luôn được xây dựng dựa trên yêu cầu của khách hàng!
- Các yêu cầu của khách hàng được phân tích thành các chức năng của ứng dụng.
- Các chức năng của ứng dụng sau đó được phân tích thành nhiều chức năng nhỏ hơn.
- Các chức năng này tiếp tục được phân tích chi tiết thành chức năng của các lớp đối tượng mà lập trình viên có thể hiện thực hóa bằng các dòng lệnh viết bằng ngôn ngữ lập trình.



Thư viện

- Trong quá trình xây dựng ứng dụng, sẽ có những chức năng mà chúng ta **sử dụng lại** rất nhiều lần trong những dự án khác nhau.
- Mã nguồn của những chức năng này sẽ được **bien dịch** thành các tập tin có đuôi mở rộng là **jar** và được sử trong những dự án khác.
- Những thư viện này có thể được **chia sẻ** rộng rãi cho cộng động lập trình viên khắp thế giới sử dụng qua Maven repository hoặc chia sẻ cả **mã nguồn** trên Github, Gitlab, Gitbucket...
- Sử dụng thư viện giúp công việc lập trình **dễ dàng** hơn, giúp **rút ngắn thời gian** phát triển ứng dụng.



Ứng dụng



- Ứng dụng thường được xây dựng dựa trên việc **sử dụng nhiều thư viện**.
- Mỗi thư viện như một **mảnh ghép** LEGO để xây nên một ứng dụng lớn và phức tạp.
- Nhưng việc sử dụng thư viện là **không bắt buộc**. Bạn có thể **tự viết tất cả** các chức năng mình cần mà không sử dụng bất kỳ thư viện bên ngoài nào.
- Việc **phát triển mọi thứ** từ đầu đòi hỏi tài nguyên rất lớn về **nhân lực, tài chính** và **thời gian**. Chính vì vậy chỉ một số công ty công nghệ lớn hiện nay mới đủ nguồn lực làm điều này, ví dụ như **Facebook, Google**.

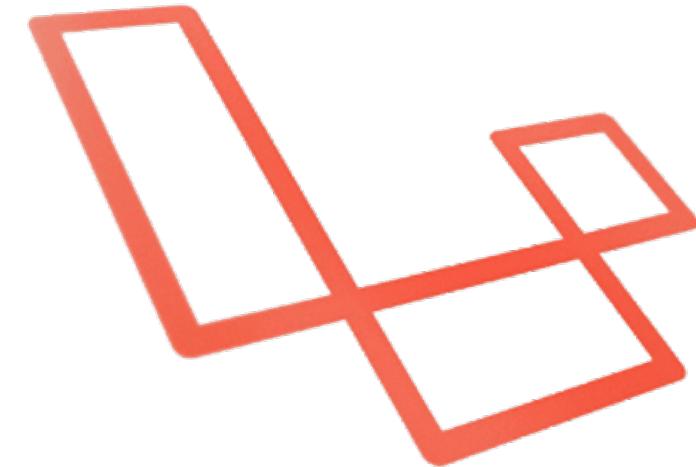
Framework



ASP.net



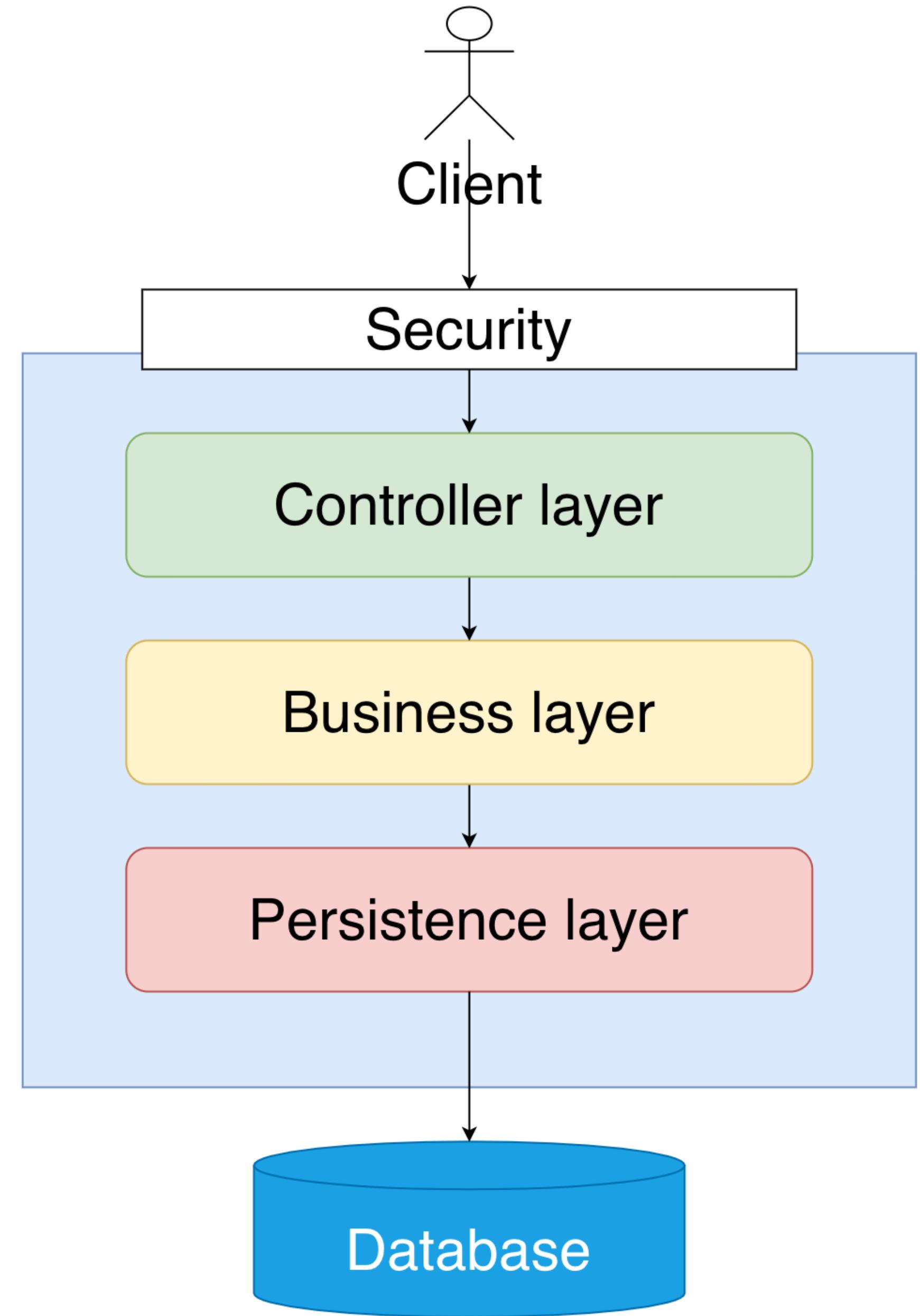
Express



laravel

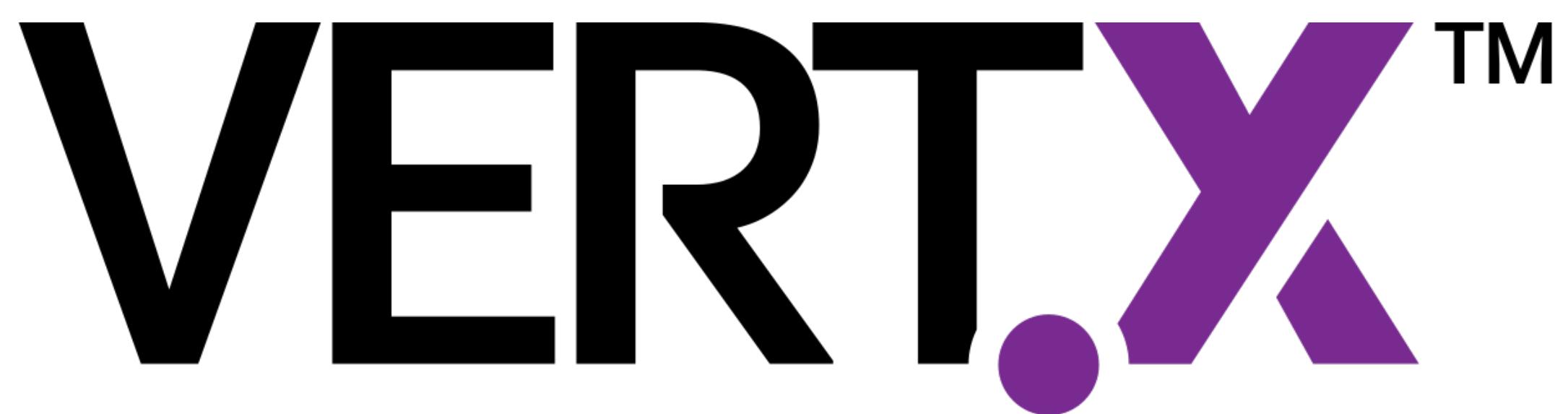


- Trong quá trình phát triển ứng dụng, chúng ta nhận thấy rất nhiều **chức năng** được **lắp đi lắp lại** rất nhiều lần.
- Để **tiết kiệm thời gian** phát triển, lập trình viên sẽ sử dụng các chức năng có sẵn từ các **thư viện** kết hợp thêm **mã nguồn riêng** để phát triển tính năng.
- Các **thư viện** đơn giản được **tập hợp** và **liên kết** lại với nhau để xây dựng các **chức năng phức tạp** hơn.



Framework là gì?

- Framework là **tập hợp** các **thư viện** được framework **liên kết** với nhau để tạo thành các **chức năng** của framework.
- Framework **cung cấp** sẵn các **tính năng** để phát triển ứng dụng, giúp việc xây dựng ứng dụng **nhanh** và **đơn giản** hơn.
- Framework được **chuẩn hóa** sẵn, giúp ứng dụng được xây dựng từ framework đáp ứng được các **tiêu chuẩn** của một ứng dụng **Enterprise**, giúp ứng dụng được vận hành và bảo trì dễ dàng hơn.



Ưu điểm khi sử dụng framework

- Framework về cơ bản **cung cấp sẵn** cho ta một **ứng dụng khung** đã hoạt động một cách **hiệu quả** và được **chuẩn hóa**. Việc này giúp lập trình viên không phải mất nhiều công sức xây dựng ứng dụng từ đầu.
- Framework giúp **giảm số lượng mã nguồn** lập trình viên phải viết đi đáng kể. Điều này giúp tiết kiệm **thời gian** và **nhân lực** rất lớn. Kết hợp với việc đã chuẩn hóa giúp quá trình **bảo trì dễ dàng** hơn.
- Framework giúp tăng tính **bảo mật** cho ứng dụng. Các chức năng chống các kiểu tấn công ứng dụng như SQL injection, cross-site request forgery, ... đã được ngăn chặn sẵn bởi framework.



Nhược điểm khi sử dụng framework

- Việc **phụ thuộc** vào framework quá nhiều sẽ làm giảm đi khả năng **hiểu sâu hơn** về ngôn ngữ, công nghệ mà lập trình viên đang sử dụng.
- Framework làm **giảm khả năng linh hoạt** trong kiến trúc của ứng dụng. Mỗi framework đều được xây dựng theo kiến trúc nhất định và rất **khó thay đổi** trong quá trình phát triển ứng dụng dựa trên framework.
- Ứng dụng của bạn sẽ khó bảo trì nếu framework mà bạn đang sử dụng **không được tiếp tục hỗ trợ** bởi tổ chức, cộng đồng đứng sau framework này.

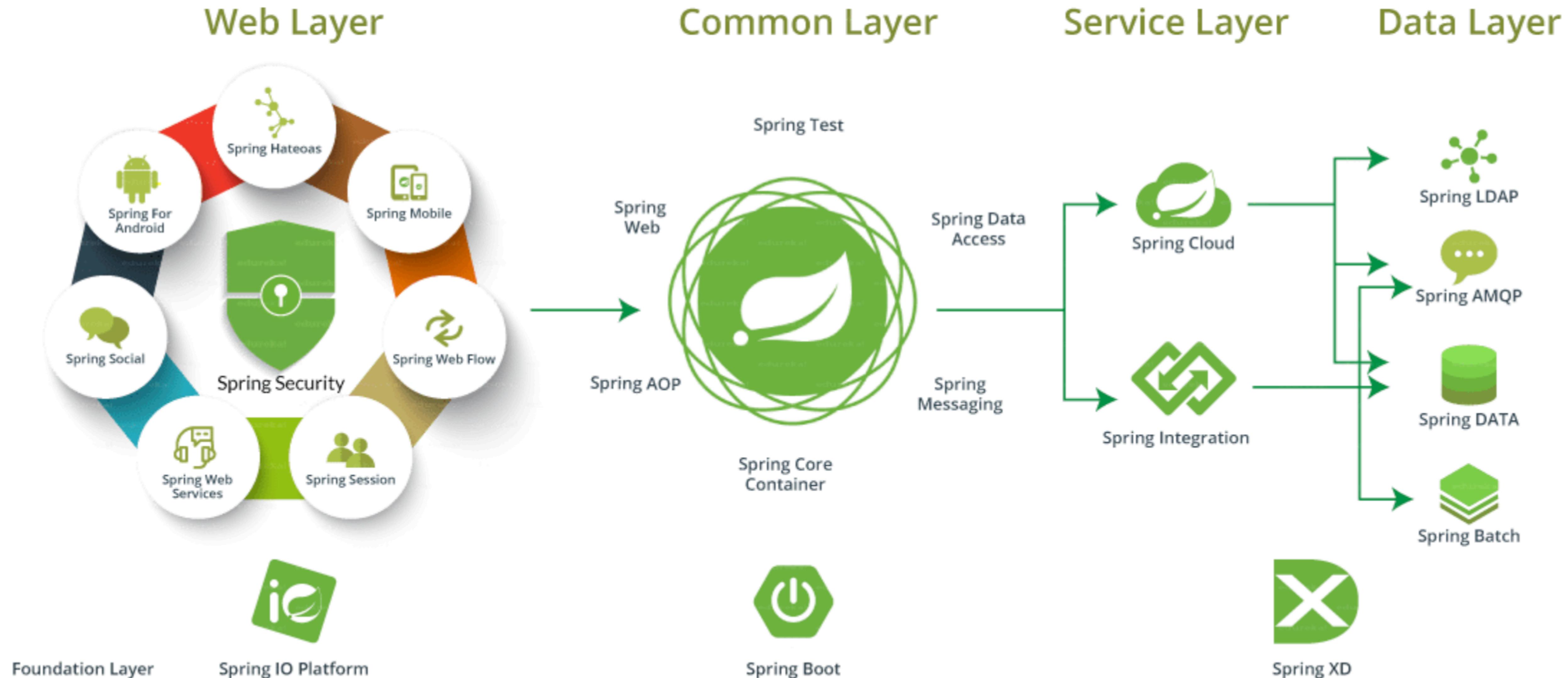




- Được giới thiệu lần đầu vào tháng mười năm 2002 bởi **Rod Johnson** và phát hành phiên bản chính thức vào tháng sáu năm 2003.
- Spring Framework là một framework **mã nguồn mở**, mọi người có thể tải về tại <https://github.com/spring-projects/spring-framework>
- Spring Framework đã trở thành **framework Java phổ biến nhất** tại thời điểm hiện tại. Trong báo cáo JVM Ecosystem Report 2020 của snyk.io, cứ mỗi 10 người được hỏi thì có 6 người trả lời là họ đang dùng Spring Framework cho công việc hàng ngày.



Spring Ecosystem





Spring Boot

It's a Kind of Magic

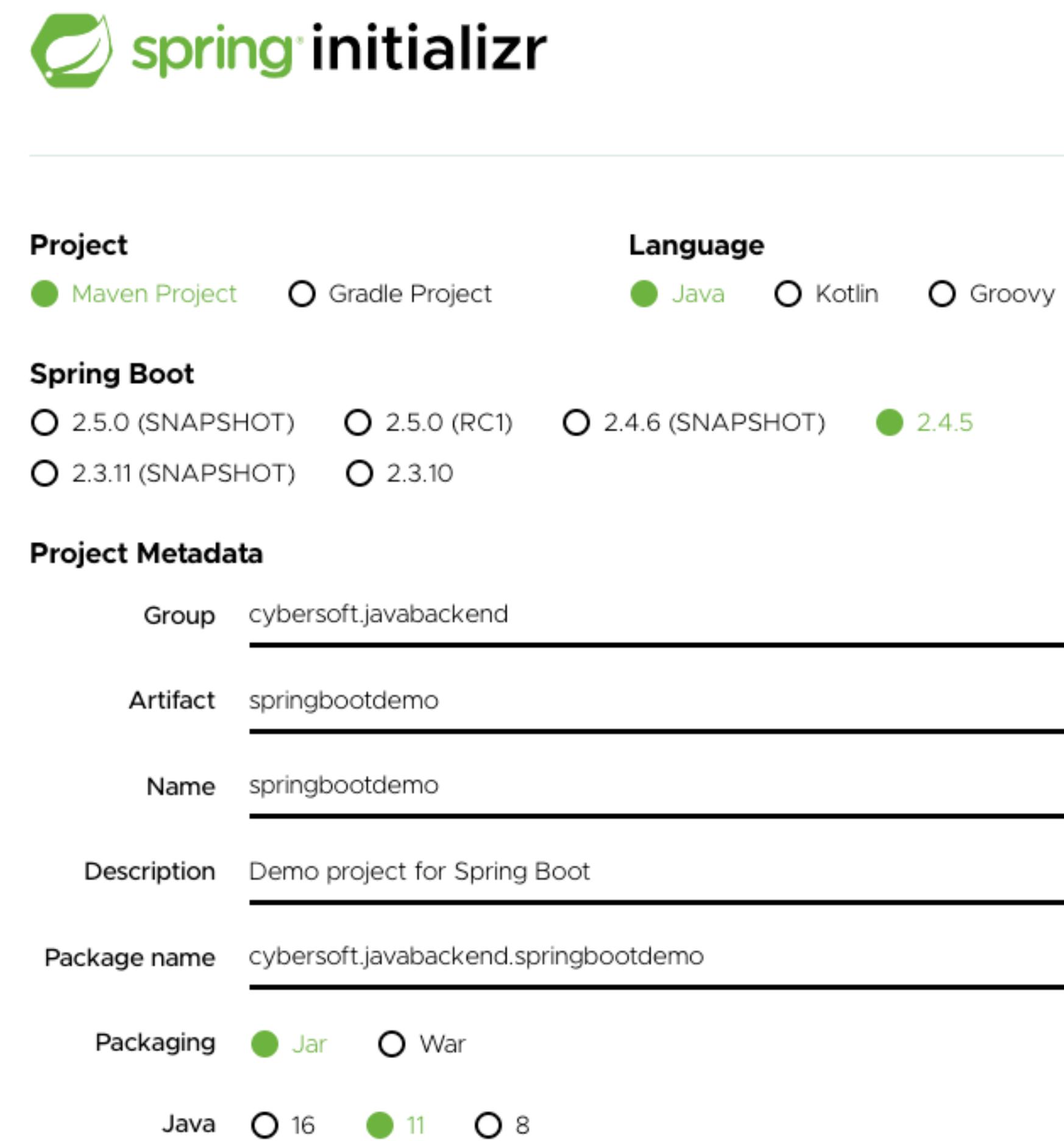
Tại sao chọn Spring Boot?

- Spring Boot có thể được biên dịch thành file jar và **chạy độc lập** như một stand-alone application.
- Cung cấp sẵn các **thư viện starter** giúp việc **cấu hình dự án** trở nên dễ dàng.
- Các module của Spring được **cấu hình** một cách **tự động** hoàn toàn. Các thư viện bên thứ cũng có thể được cấu hình tự động nếu được hỗ trợ.
- Các chức năng **metrics, health checks** được cấu hình sẵn sàng cho môi trường **production**.



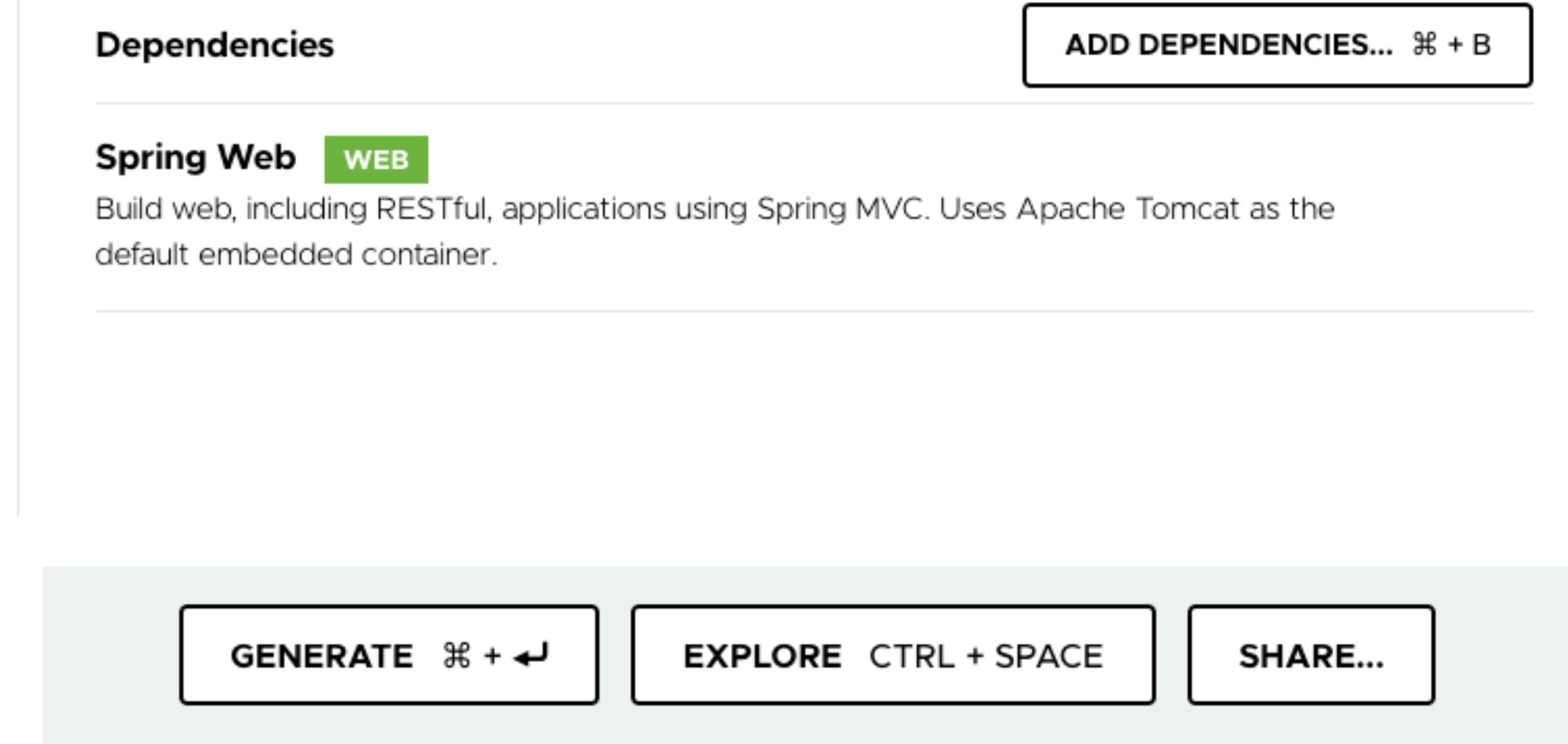
Tạo một dự án Spring Boot

- Vào trang <https://start.spring.io/>
- **Project**: chọn build tool là Maven hoặc Gradle để sử dụng cho dự án
- **Language**: ba ngôn ngữ được hỗ trợ là Java, Kotlin và Groovy
- **Spring Boot**: các bạn chọn phiên bản 2.4.5
- **Project Metadata**: các thông tin về dự án, các bạn có thể điền như hình hoặc theo domain mình đang sử dụng.



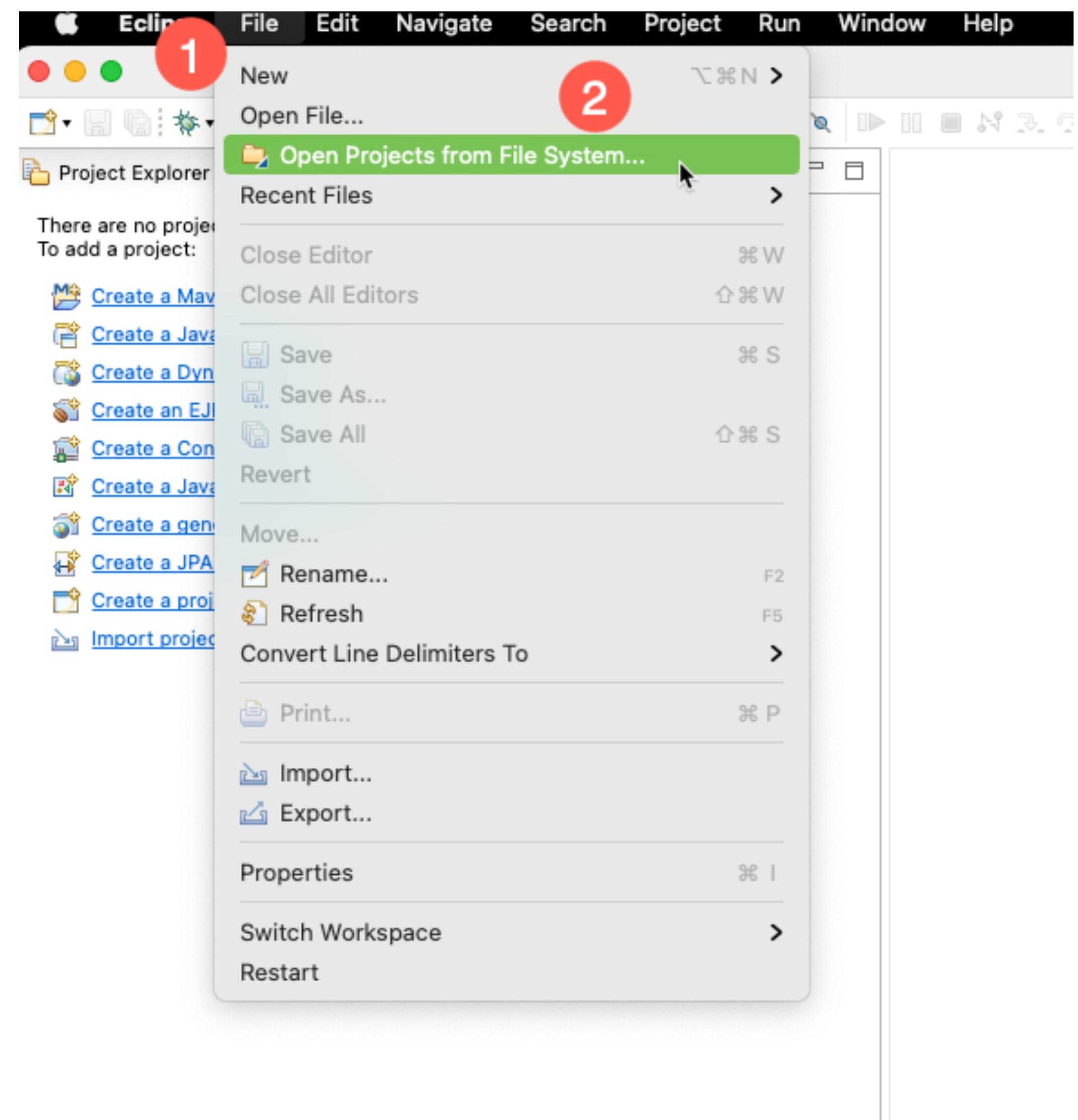
Tạo một dự án Spring Boot

- Dependencies: các bạn chọn Spring Web.
- GENERATE: tải về mã nguồn của dự án.
- EXPLORE: xem trước các tổ chức thư mục của dự án.
- SHARE: gửi cấu hình dự án đến cho bạn bè, đồng nghiệp.



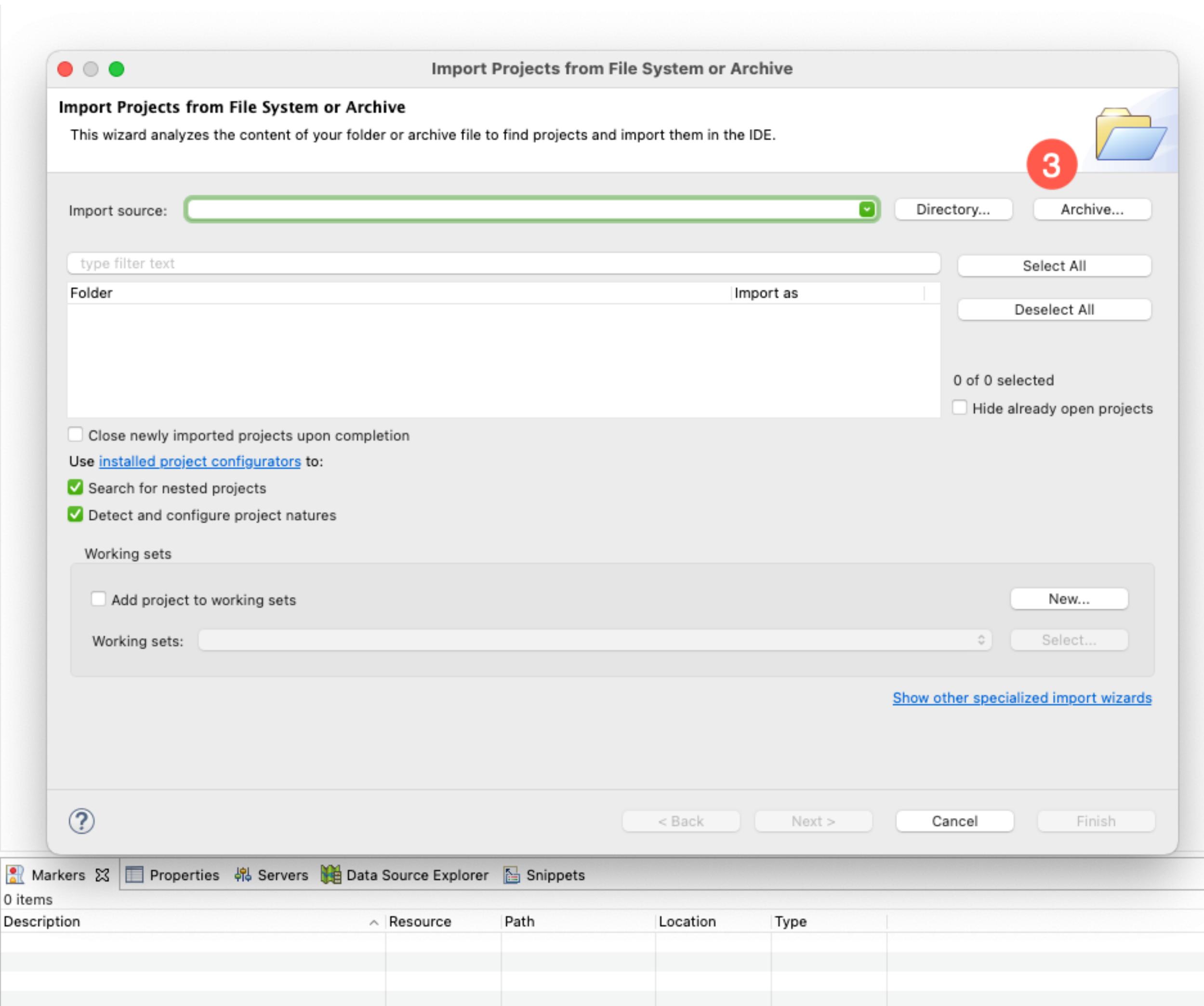
Tạo một dự án Spring Boot

- Trên thanh menu chọn File -> Open Projects from File System



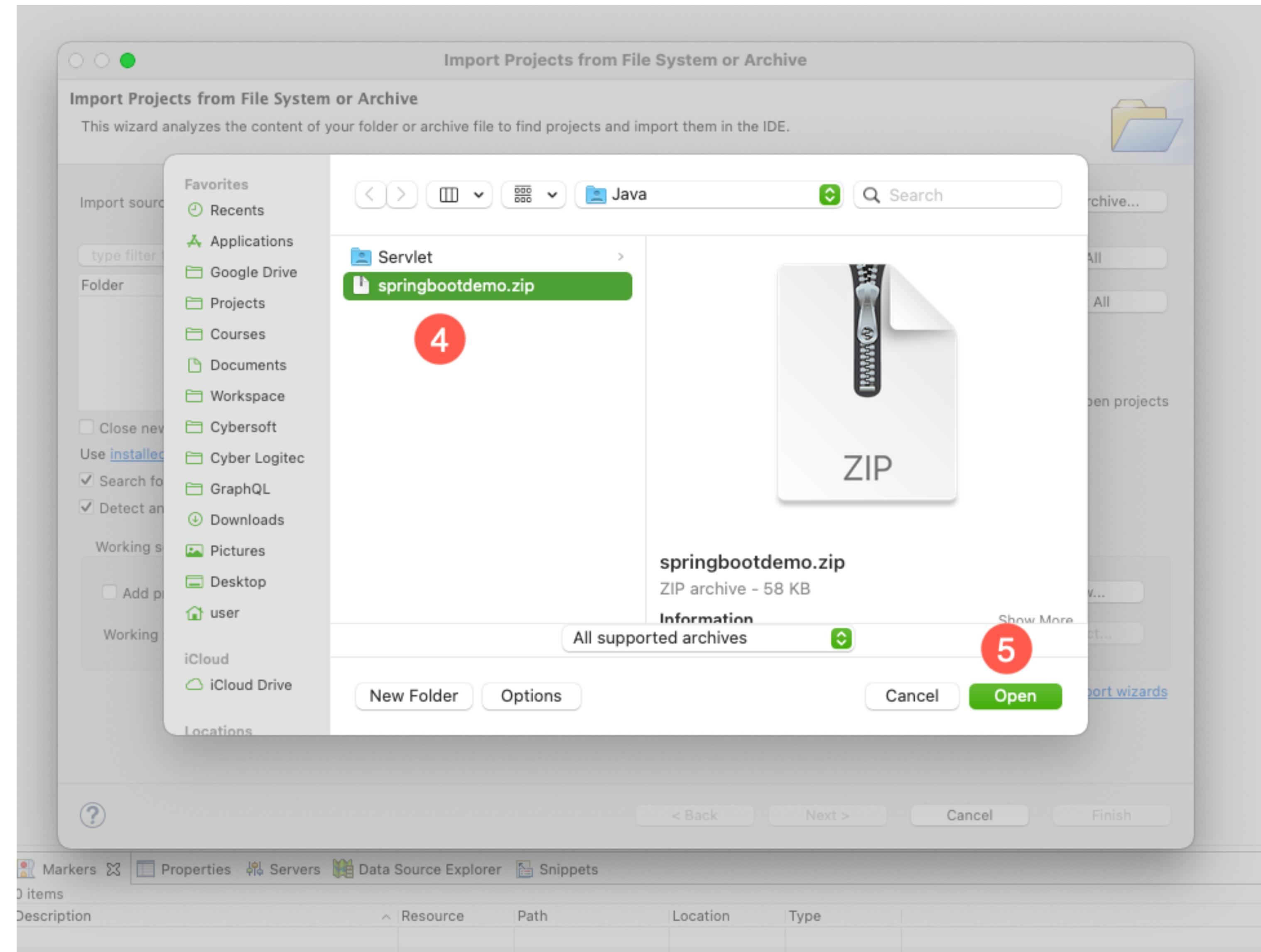
Tạo một dự án Spring Boot

- Click chọn nút Archive



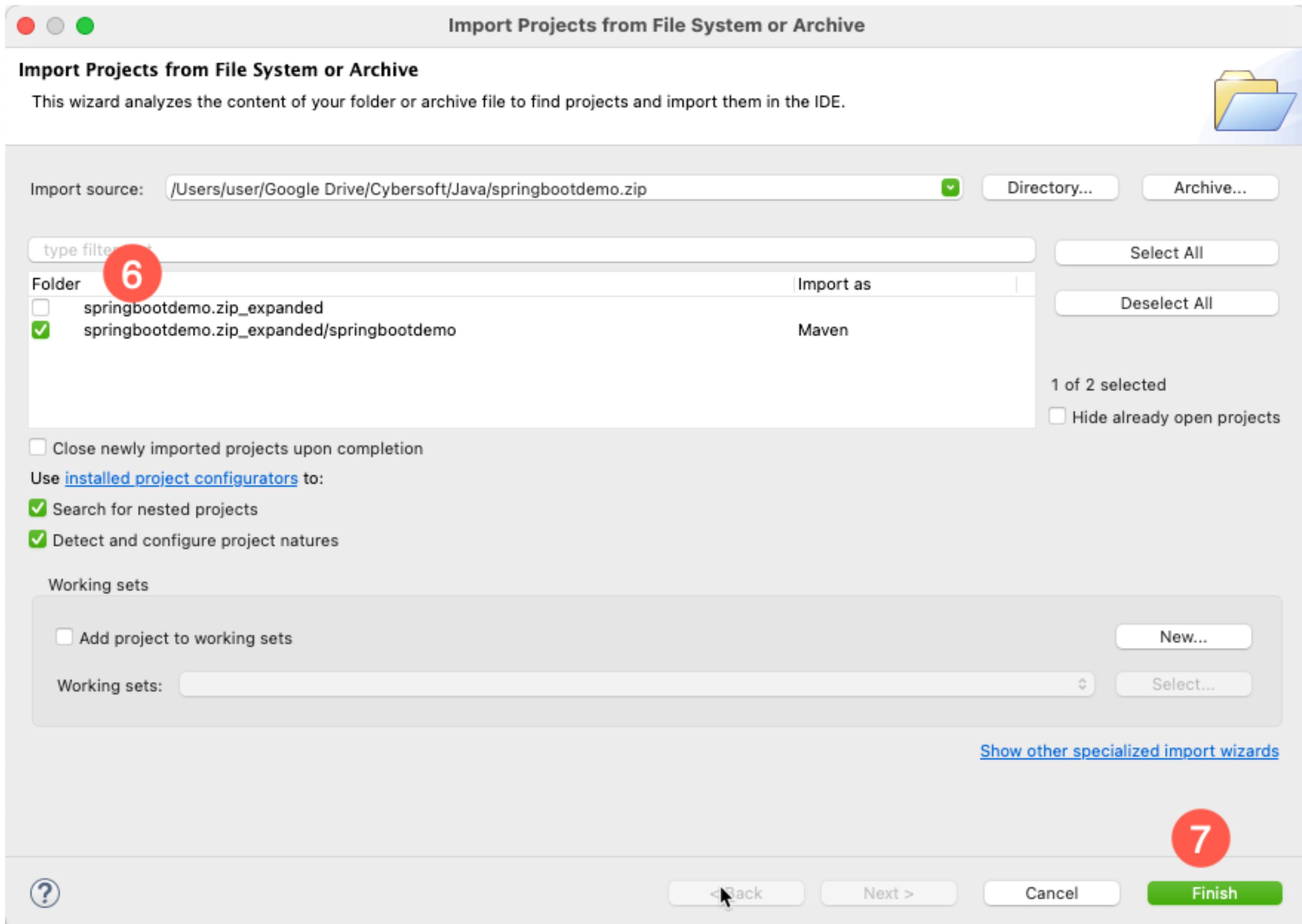
Tạo một dự án Spring Boot

- Chọn file **zip** của dự án được tải về khi ấn nút "GENERATE"
- Ấn nút **Open**



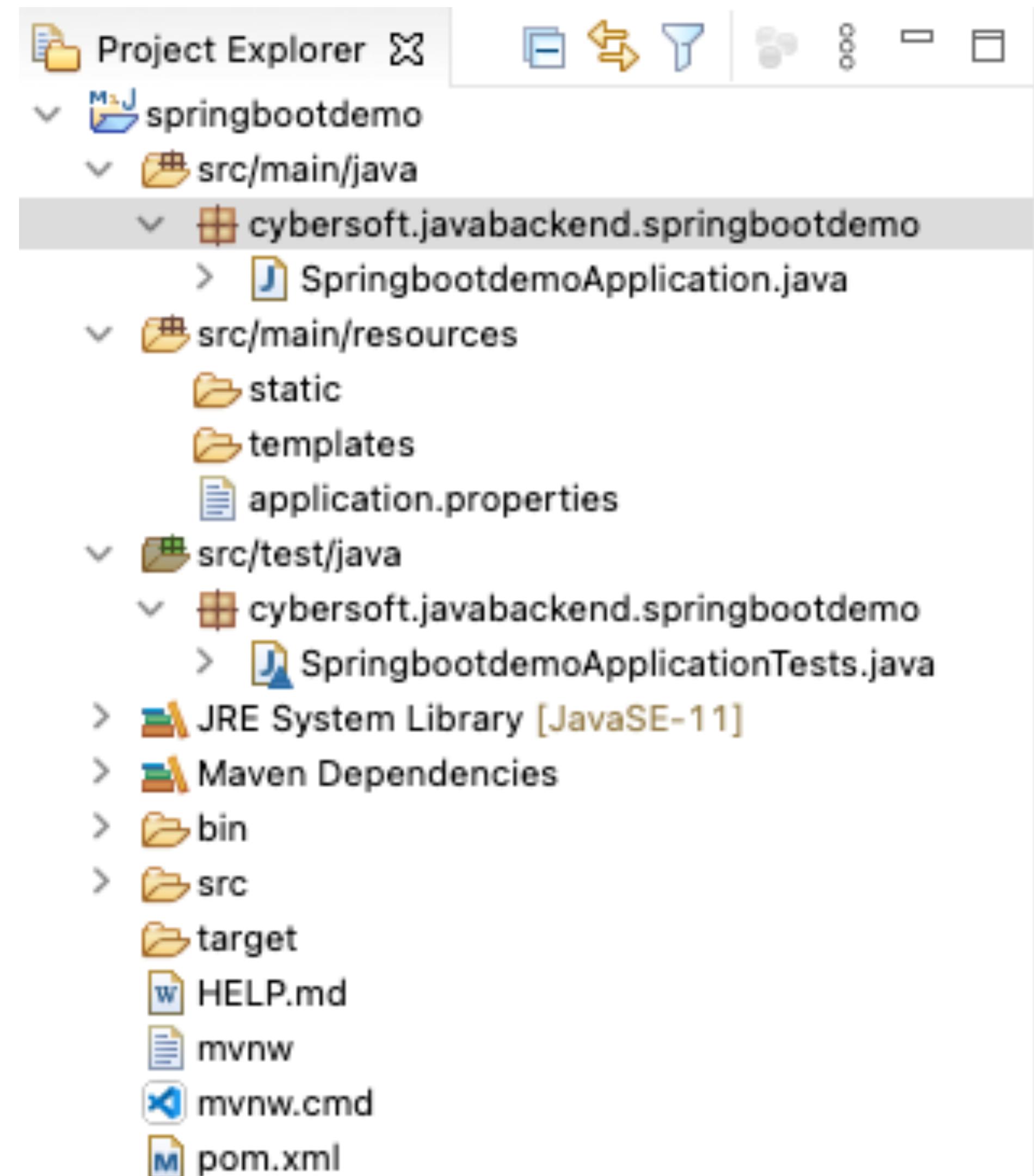
Tạo một dự án Spring Boot

- Bỏ chọn thư mục không được Import as Maven
- Án nút Finish để tiến hành import project vào workspace hiện tại.



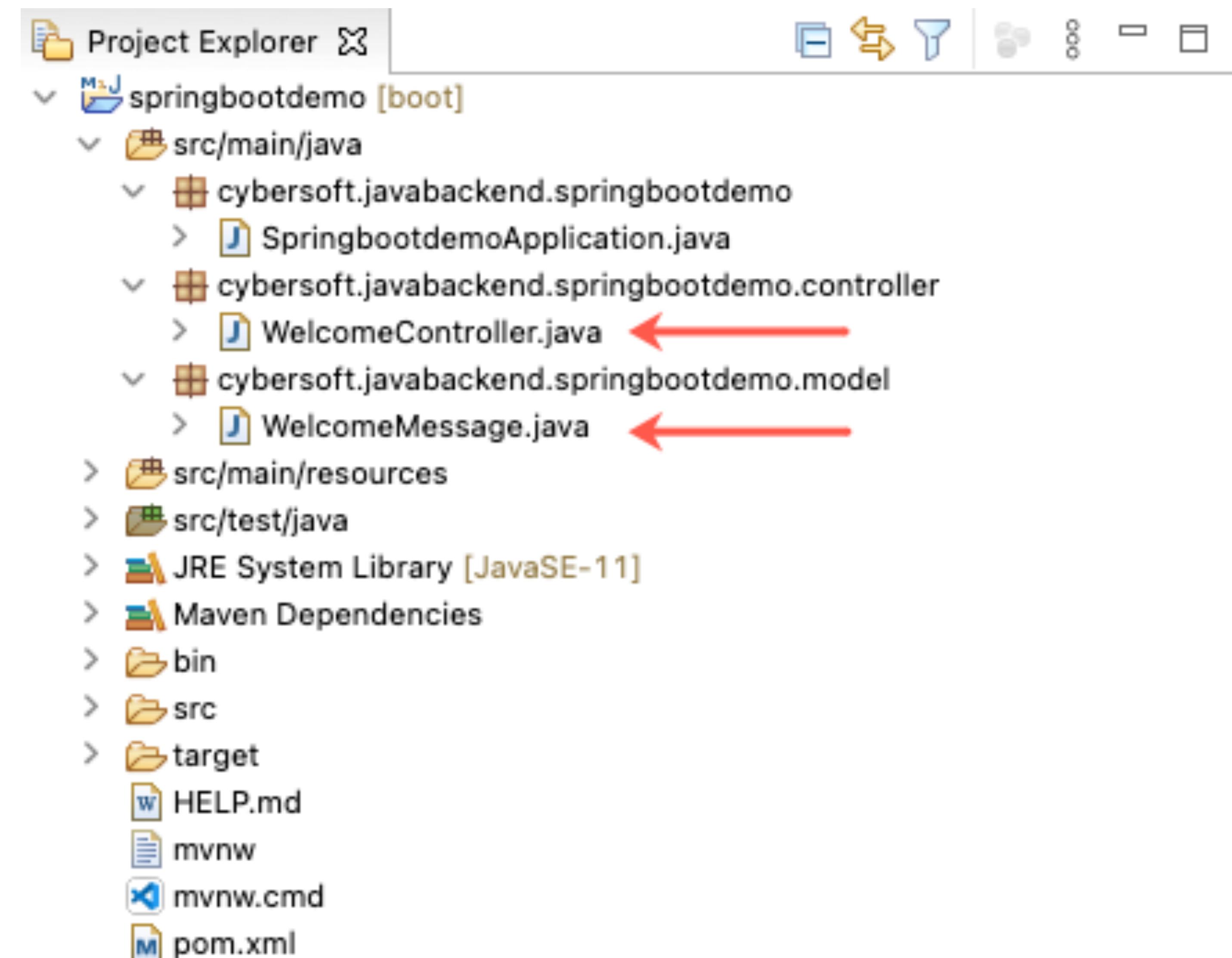
Tạo một dự án Spring Boot

- Sau khi Eclipse hoàn tất xong việc tải thư viện cho Spring Boot, chúng ta sẽ được project có cấu trúc thư mục như trong hình.
- **src/main/java**: chứa tất cả những package và file class java của ứng dụng.
- **src/main/resources**: chứa các tài nguyên html, css, js, assets khác và các file cấu hình
- **src/test/java**: chứa các cái file test của chương trình
- **pom.xml**: file cấu hình của dự án Maven

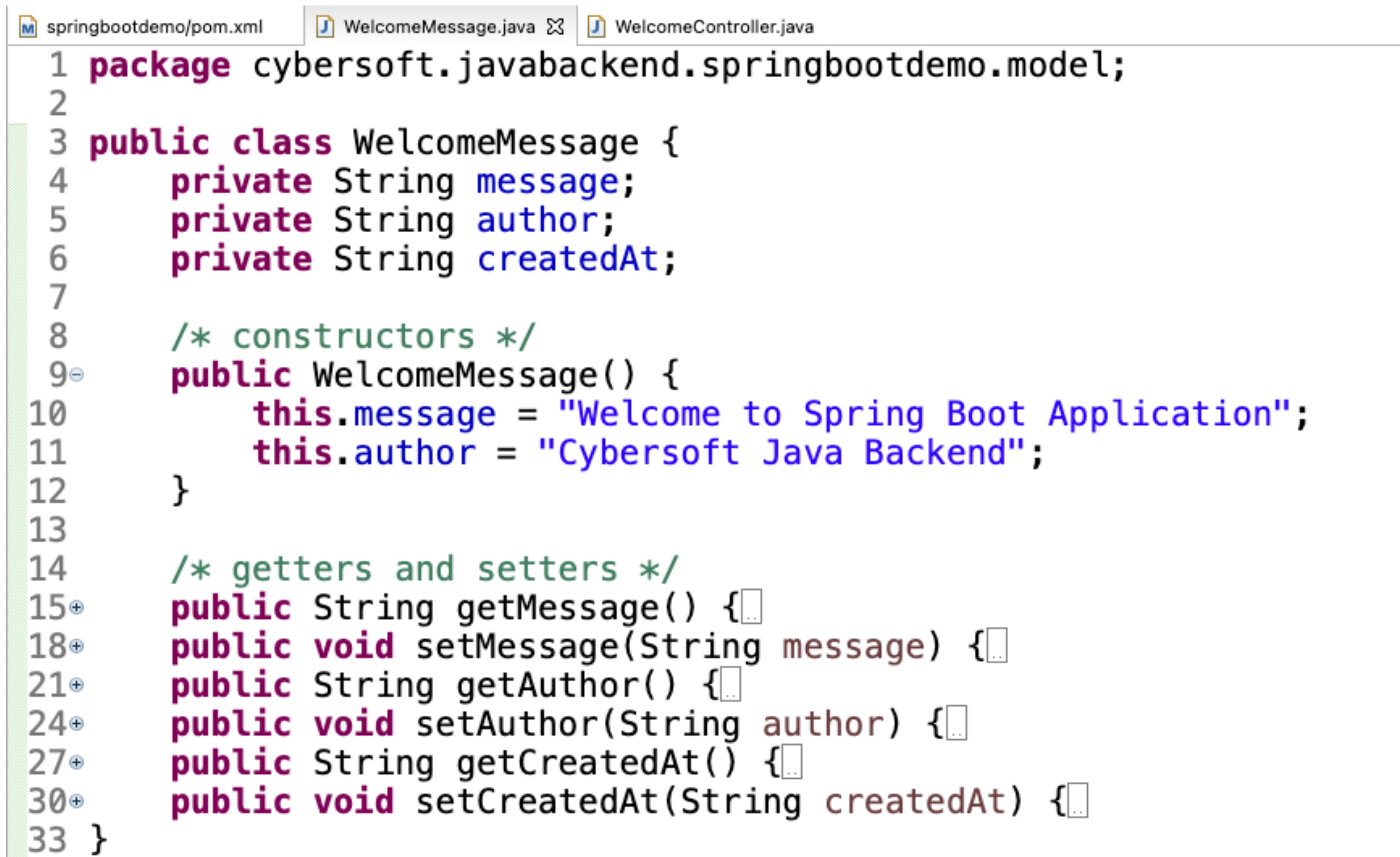


Tạo một dự án Spring Boot

- Tạo package model và package controller
- Tạo 2 file WelcomeMessage và WelcomeController như hình bên.



Tạo một dự án Spring Boot

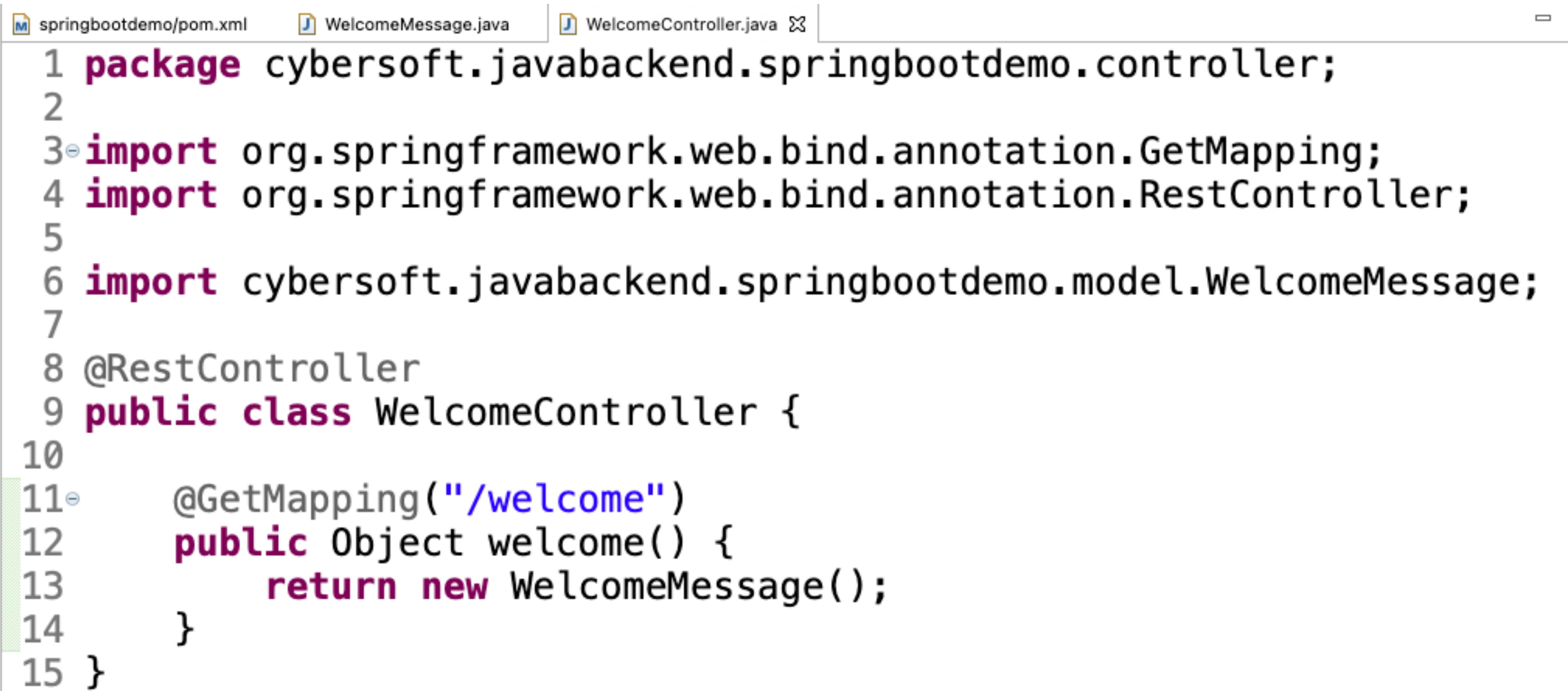


The screenshot shows a Java code editor with the following details:

- Project navigation bar: springbootdemo/pom.xml, WelcomeMessage.java, WelcomeController.java
- Code content:

```
1 package cybersoft.javabackend.springbootdemo.model;
2
3 public class WelcomeMessage {
4     private String message;
5     private String author;
6     private String createdAt;
7
8     /* constructors */
9     public WelcomeMessage() {
10         this.message = "Welcome to Spring Boot Application";
11         this.author = "Cybersoft Java Backend";
12     }
13
14     /* getters and setters */
15     public String getMessage() {
16     }
17     public void setMessage(String message) {
18     }
19     public String getAuthor() {
20     }
21     public void setAuthor(String author) {
22     }
23     public String getCreatedAt() {
24     }
25     public void setCreatedAt(String createdAt) {
26     }
27 }
```

Tạo một dự án Spring Boot

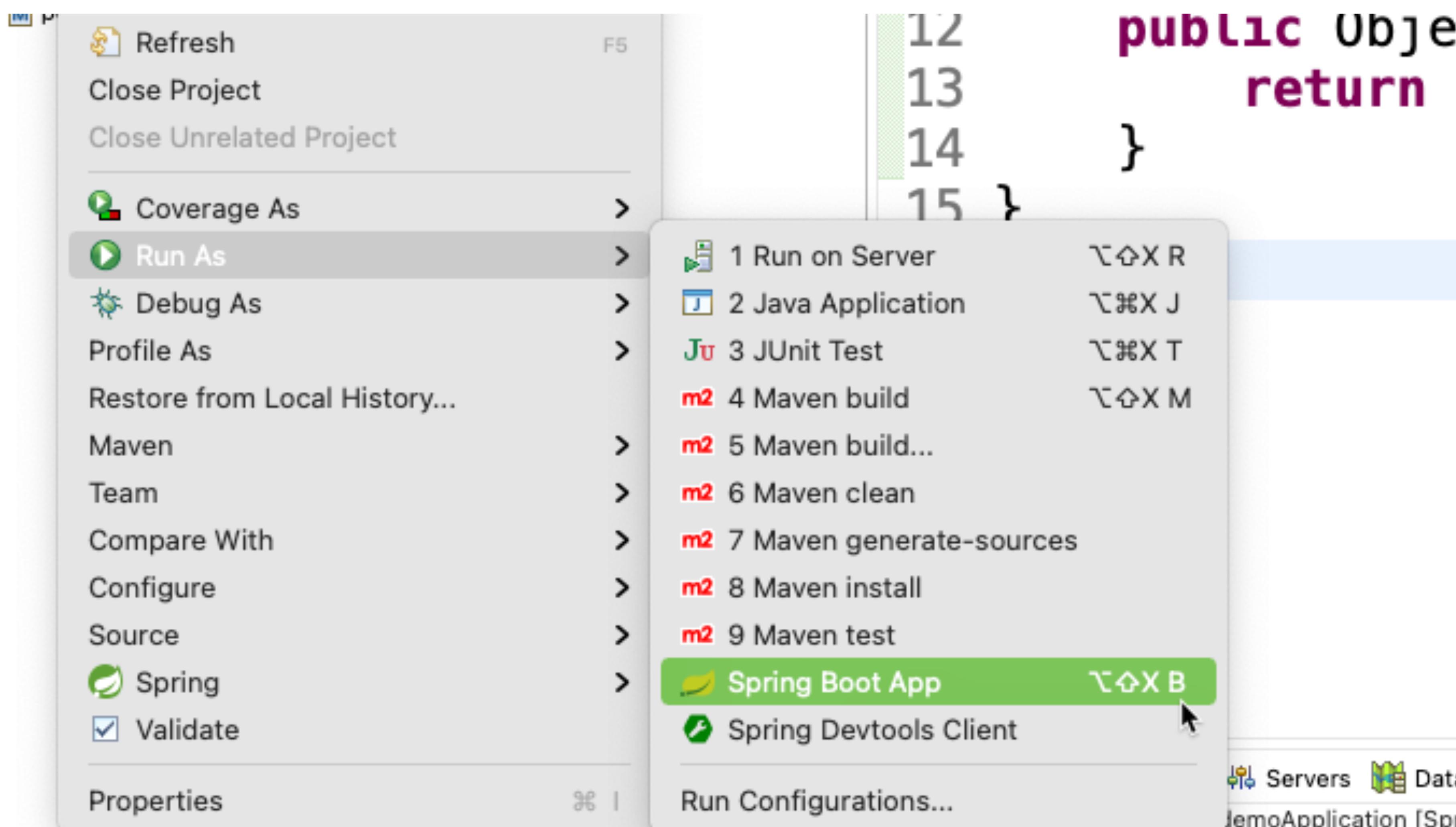


The screenshot shows a Java code editor with three tabs at the top: 'springbootdemo/pom.xml', 'WelcomeMessage.java', and 'WelcomeController.java'. The 'WelcomeController.java' tab is active, displaying the following code:

```
1 package cybersoft.javabackend.springbootdemo.controller;
2
3 import org.springframework.web.bind.annotation.GetMapping;
4 import org.springframework.web.bind.annotation.RestController;
5
6 import cybersoft.javabackend.springbootdemo.model.WelcomeMessage;
7
8 @RestController
9 public class WelcomeController {
10
11     @GetMapping("/welcome")
12     public Object welcome() {
13         return new WelcomeMessage();
14     }
15 }
```

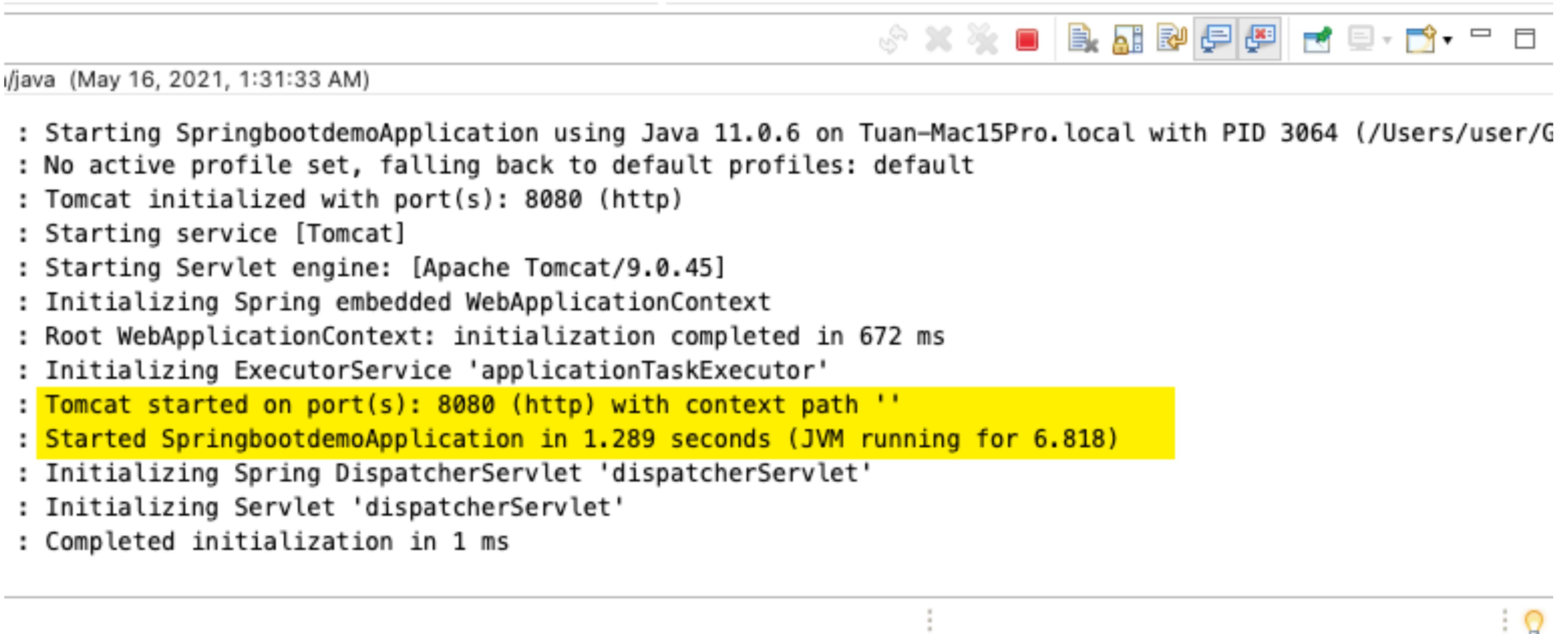
Tạo một dự án Spring Boot

- Chạy ứng dụng Spring Boot: click phải project Spring Boot -> Run As -> Spring Boot App



Tạo một dự án Spring Boot

- Ứng dụng Spring Boot đã chạy thành công.

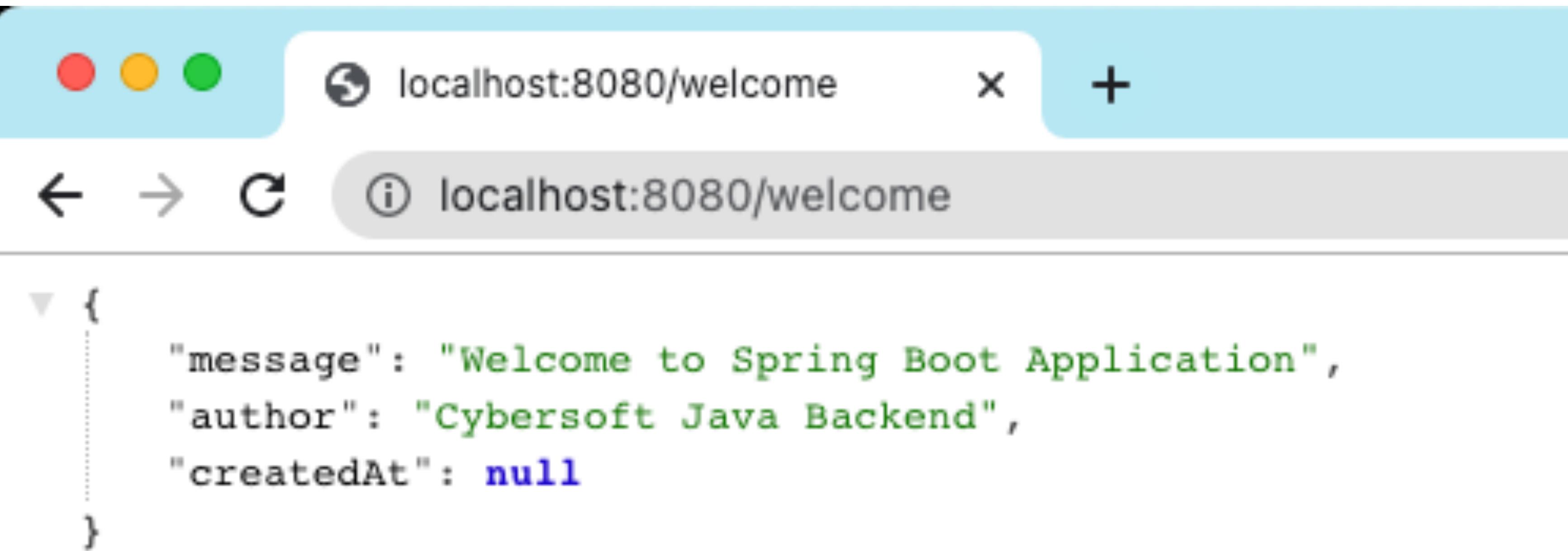


The screenshot shows a Java application window with a toolbar at the top and a console output area below. The toolbar icons include: file (new, open, save, close), edit (undo, redo), search (find, replace), and others. The console output window has tabs at the top labeled 'java' (selected) and 'May 16, 2021, 1:31:33 AM'. The main area displays the log output of a Spring Boot application starting up. A yellow highlight box surrounds the lines indicating the application has started successfully:

```
: Starting SpringbootdemoApplication using Java 11.0.6 on Tuan-Mac15Pro.local with PID 3064 (/Users/user/G
: No active profile set, falling back to default profiles: default
: Tomcat initialized with port(s): 8080 (http)
: Starting service [Tomcat]
: Starting Servlet engine: [Apache Tomcat/9.0.45]
: Initializing Spring embedded WebApplicationContext
: Root WebApplicationContext: initialization completed in 672 ms
: Initializing ExecutorService 'applicationTaskExecutor'
: Tomcat started on port(s): 8080 (http) with context path ''
: Started SpringbootdemoApplication in 1.289 seconds (JVM running for 6.818)
: Initializing Spring DispatcherServlet 'dispatcherServlet'
: Initializing Servlet 'dispatcherServlet'
: Completed initialization in 1 ms
```

Tạo một dự án Spring Boot

- Mở web browser bất kỳ, vào địa chỉ <http://localhost:8080/welcome>



```
{  
    "message": "Welcome to Spring Boot Application",  
    "author": "Cybersoft Java Backend",  
    "createdAt": null  
}
```