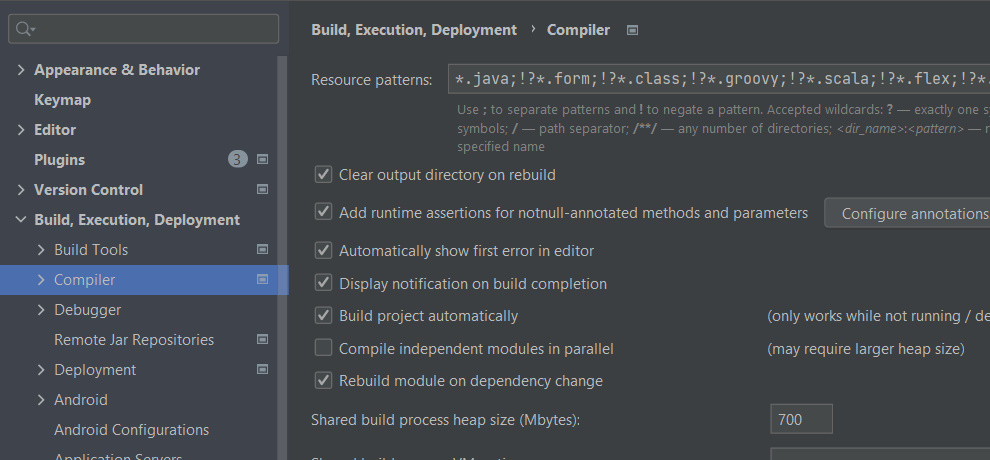
**AutoUpdate**

* Daca facem vreo oarecare modificatie la codul nostru, trebuie sa dam mereu restart la aplicatia noastra, si asta cam e problema.
* Aici ne poate ajuta spring-boot-devtools
* Acesta va face ca aplicatia sa fie restartata mereu cand codul nostru e actualizat, adica da auto reloading
* Trebuie sa adaugam o dependenta in pom.xml si inca ceva sa mai facem

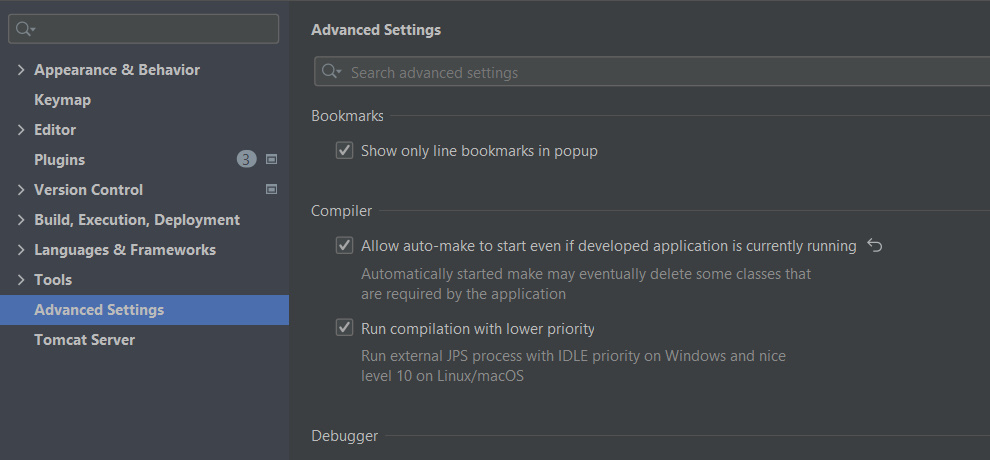


<dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-devtools</artifactId>  
</dependency>

Dupa asta facem urmatoarele:

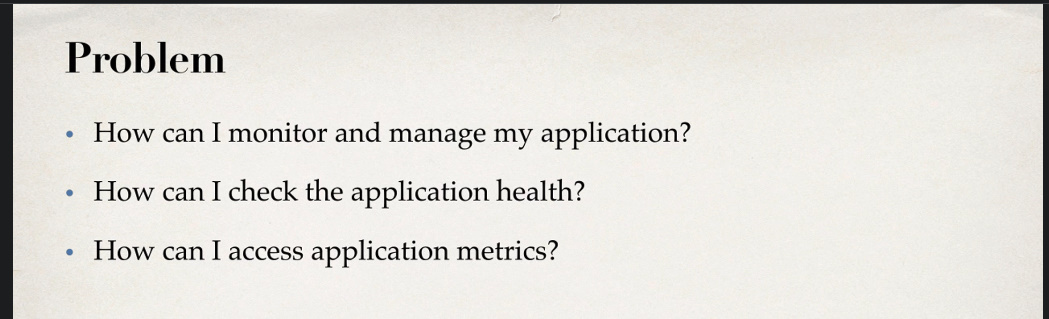




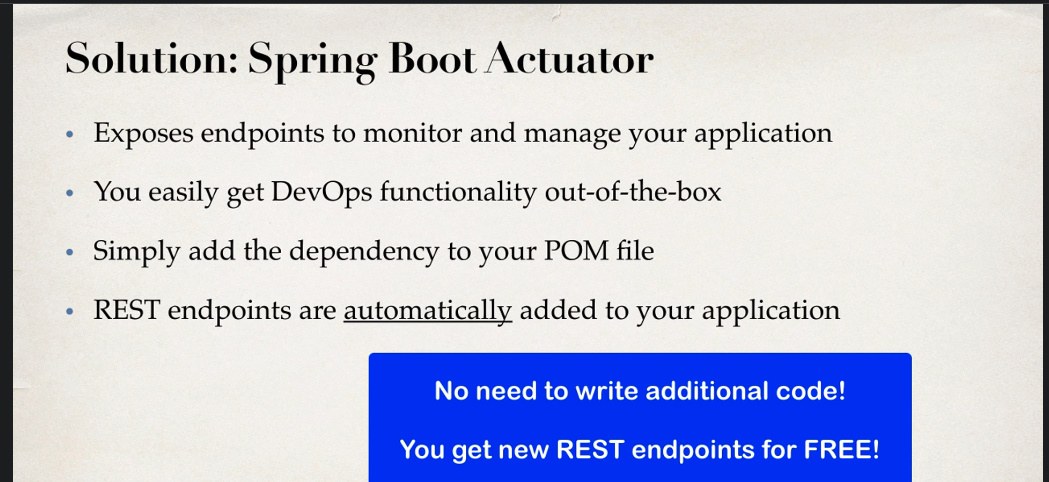




**Spring Boot Actuator**

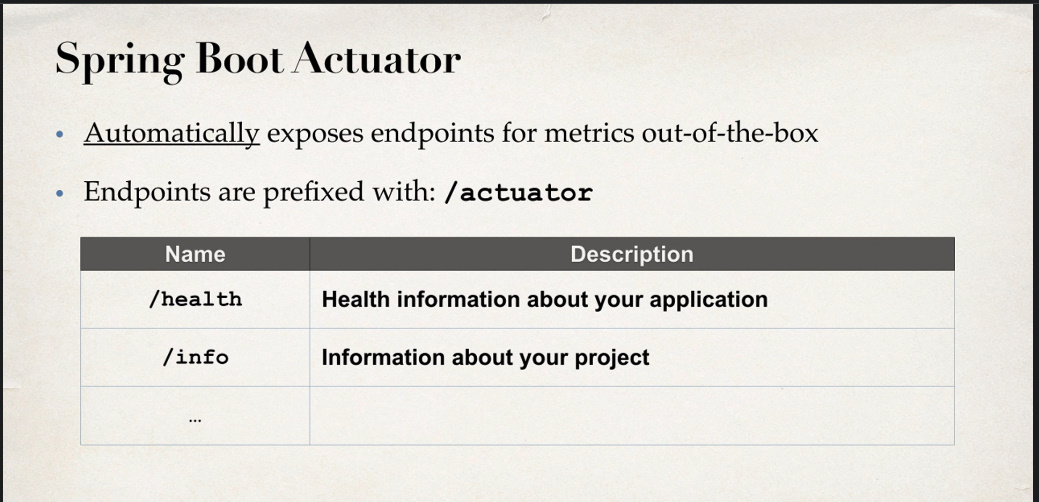


**endpoint** – rest request care ne arata date despre aplicatie

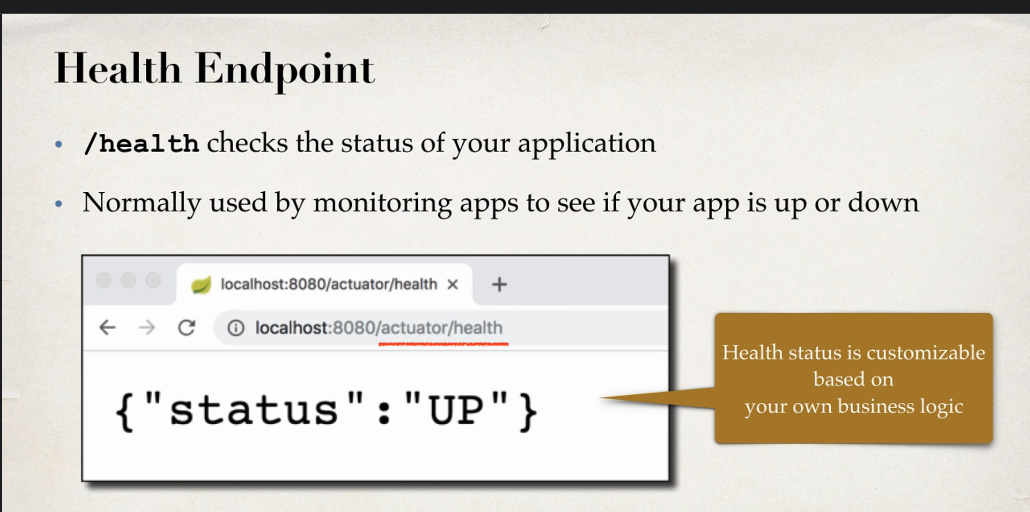


* Avem nevoie de dependenta:

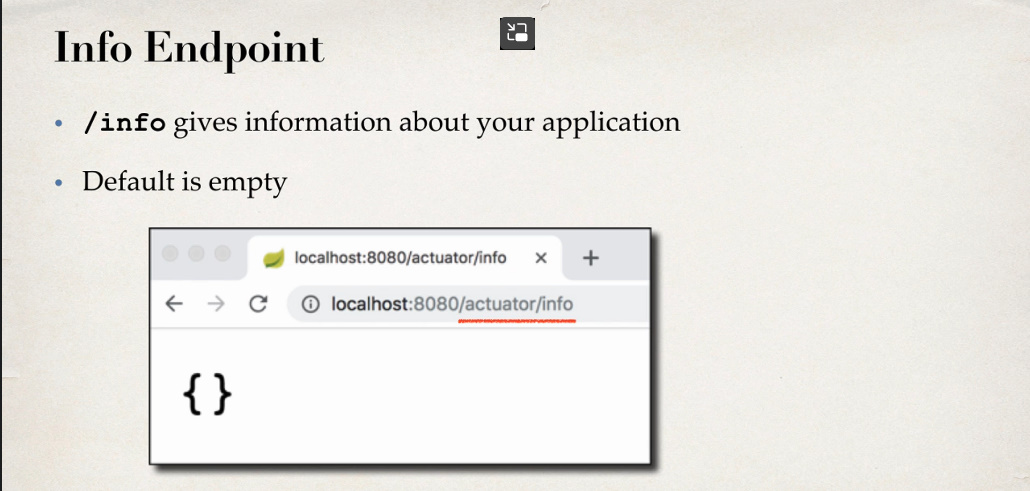
<dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-actuator</artifactId>  
</dependency>



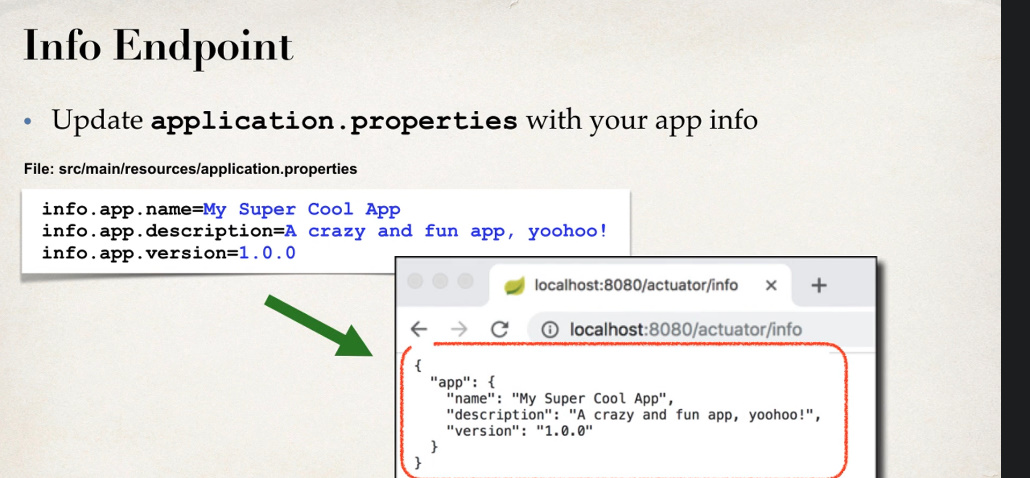
* /health



* /info



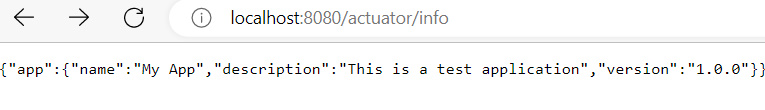
Default, acest endpoint nu returneaza nimic, caci trebuie sa setam noi ceva.



Orice proprietate care incepe cu info va fi afisat cu /info

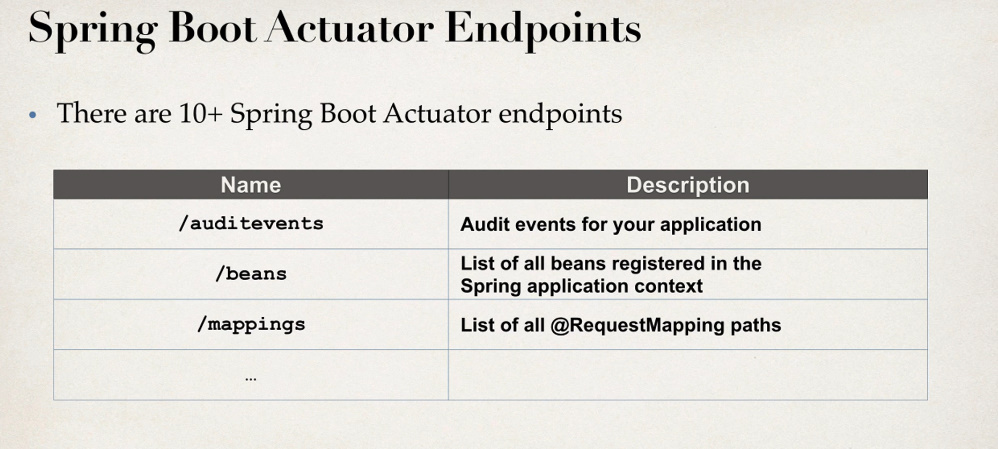
Noi alegem ce nume vrem dupa **info.** macar si info.azaza.text. De ex:

info.app.name = My App  
info.app.description = This is a test application  
info.app.version = 1.0.0

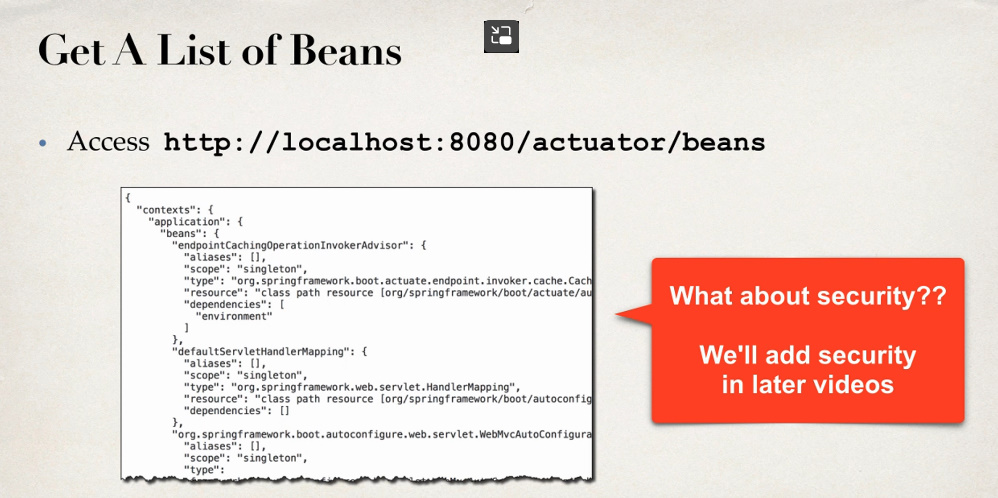


Datele arata cam urat, dar putem instala in browser un plugin numit JSON formatter ca sa fie mai frumoase

**Other endpoints**



<https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/#actuator.endpoints>



* **threaddump –** arata toate threadurile din aplicatie

**Activate endpoints**

* Endpoints, incepand cu Spring Boot 2.7, nu mai sunt activate automat, caci unele erau.
* Unicul activat automat e /health
* Acum trebuie sa le activam manual ca sa fie disponibile si sa le facem vizibile pentru HTTP requests.



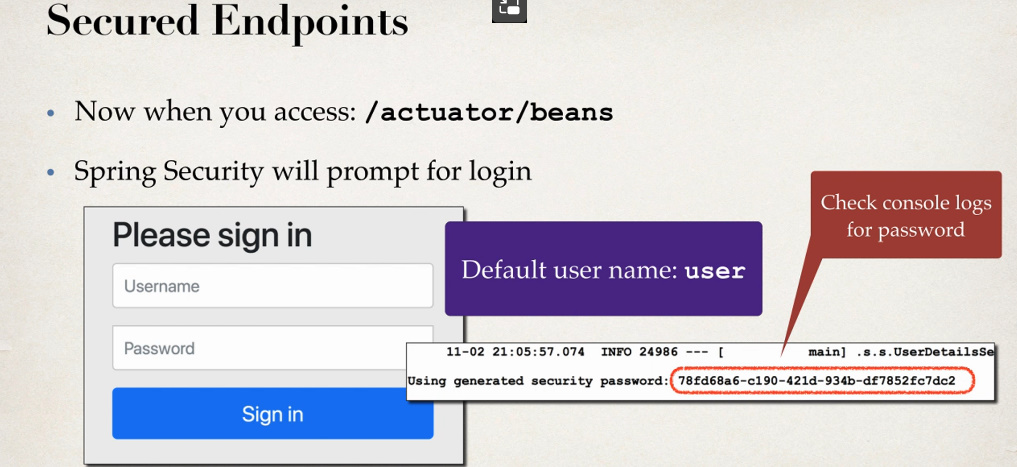
management.endpoints.web.exposure.include – face aceste endpoints disponibile in HTTP request

management.info.env.enabled = true : spune ca pentru info se vor adauga noi proprietati in fisierul .properties

management.endpoints.web.exposure.include=\*

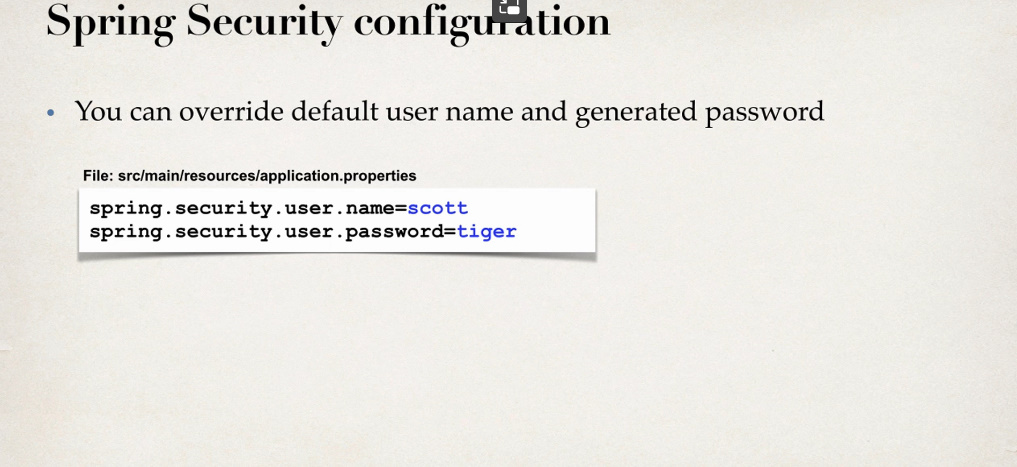
Asa facem ca toate endpoints sa fie publice in HTTP request

**Actuator Security**

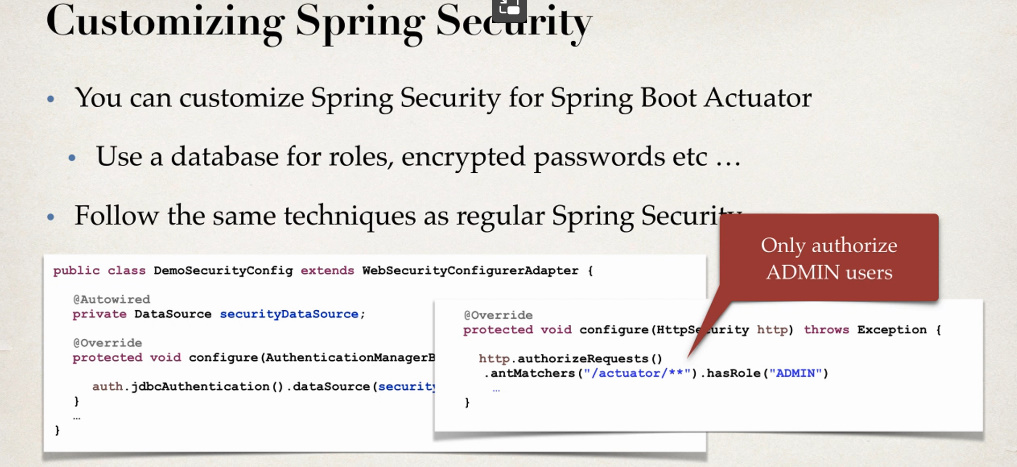
* Nu e prea bine ca oricine poate accesa endpoints
* Spring Security aici ne ajuta.
* 

Acum, cand cineva va vrea sa acceseze endpoints setate de noi, Spring Security va cere sa ne logam, insa el va genera o parola in consola la server si anume pe ea trebuie sa o punem. Username e default user. Spring Security activeaza asta automat pentru actuator!!!

* Sau, putem noi sa cream un username si o parola prin adaugarea unor proprietati in .properties file:



* Sau putem folosi metoda clasica in Spring Security, adica sa luam userii din baza de date, sa setam roluri si in dependenta de roluri, sa setam cine are acces la endpoints ori la /actuator:



* Putem si sa excludem activarea lui health prin:

management.endpoints.web.exposure.exclude = health

Sau putem sa excludem altele, daca tot am folosint include = \*

Dependency pentru Spring Security:

<dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-security</artifactId>  
</dependency>

**Pentru ca tabelul actuators din Intellij sa mearga**

spring.jmx.enabled = true  
management.endpoints.jmx.exposure.include=\*  
management.endpoints.jmx.unique-names=true