

Security - Hands-On 3

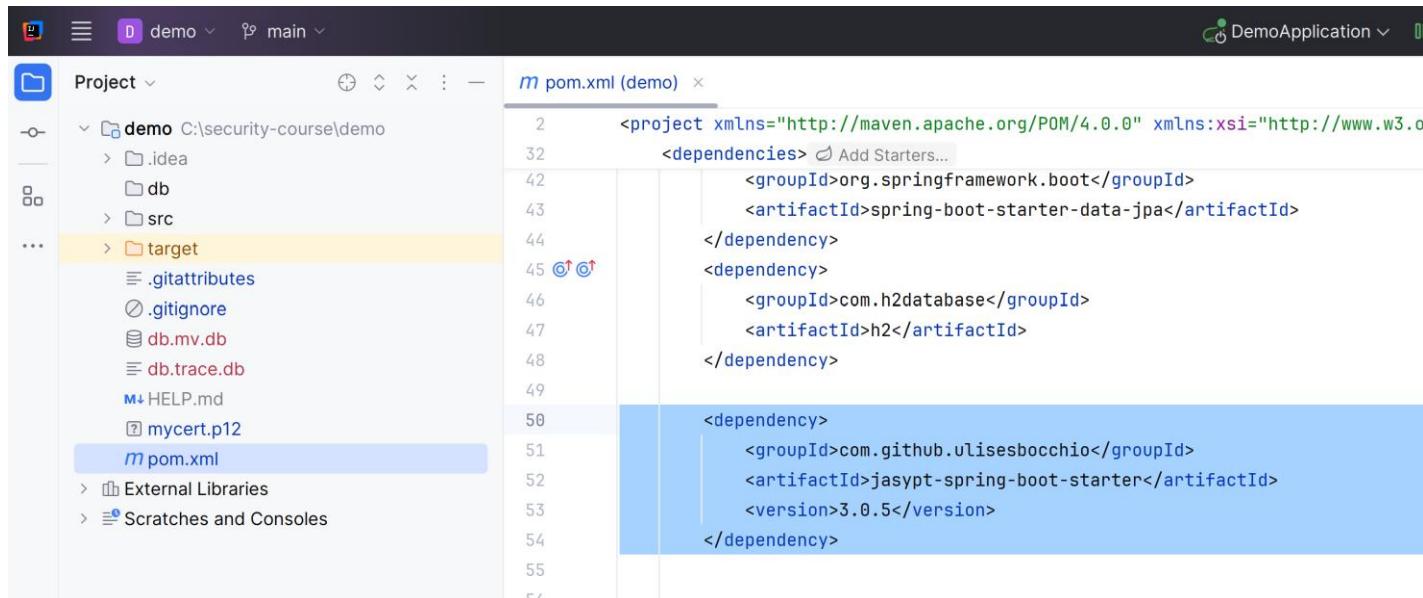
springboot credential configuration (Jasypt, spring cloud, Vault)

Esilv 2025

arnaud.nauwynck@gmail.com

Using Jasypt springboot to encrypt
configuration properties

Jasypt [1/7] : adding maven pom.xml dependency



The screenshot shows the IntelliJ IDEA interface with the project 'demo' selected. The 'pom.xml' file is open in the editor. The code in the editor is:

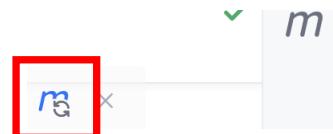
```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.example</groupId>
  <artifactId>DemoApplication</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <dependencies>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-data-jpa</artifactId>
    </dependency>
    <dependency>
      <groupId>com.h2database</groupId>
      <artifactId>h2</artifactId>
    </dependency>
    <dependency>
      <groupId>com.github.ulisesbocchio</groupId>
      <artifactId>jasypt-spring-boot-starter</artifactId>
      <version>3.0.5</version>
    </dependency>
  </dependencies>

```

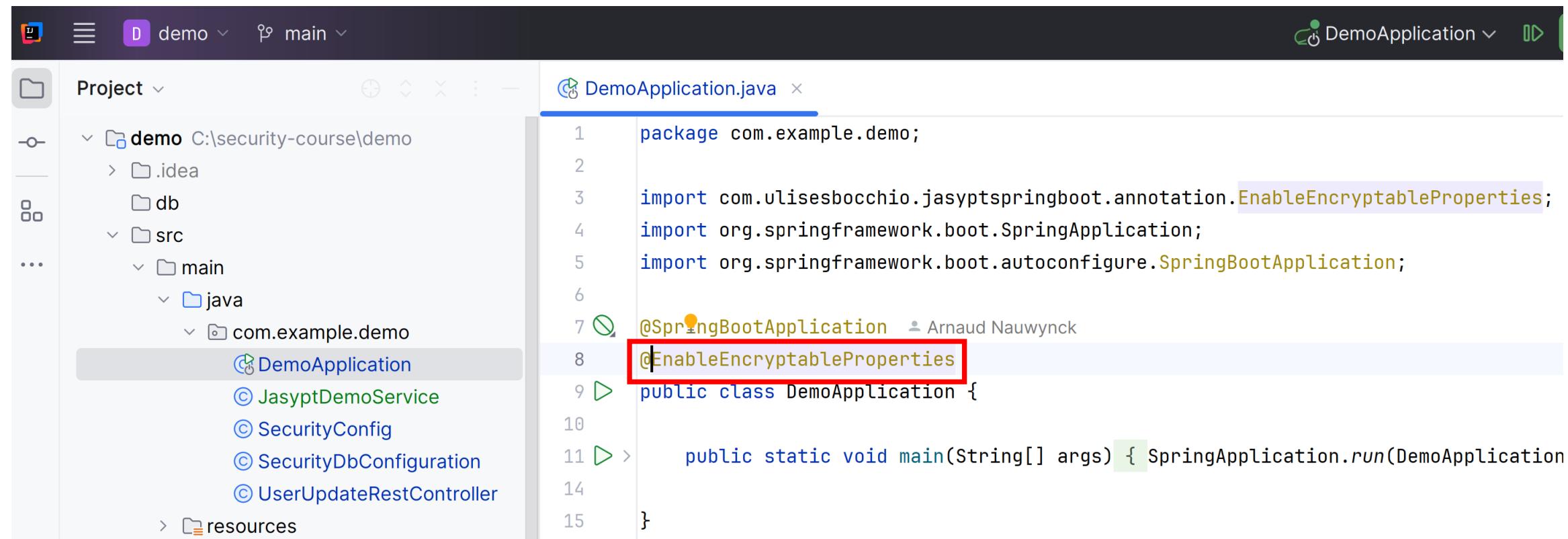
edit pom.xml, add in <dependencies>..

then click on Refresh maven

```
<dependency>
  <groupId>com.github.ulisesbocchio</groupId>
  <artifactId>jasypt-spring-boot-starter</artifactId>
  <version>3.0.5</version>
</dependency>
```



Jasypt [2/7] : Configure App to use EncryptableProperties



```
package com.example.demo;

import com.ulisesbocchio.jasyptspringboot.annotation.EnableEncryptableProperties;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
@EnableEncryptableProperties
public class DemoApplication {

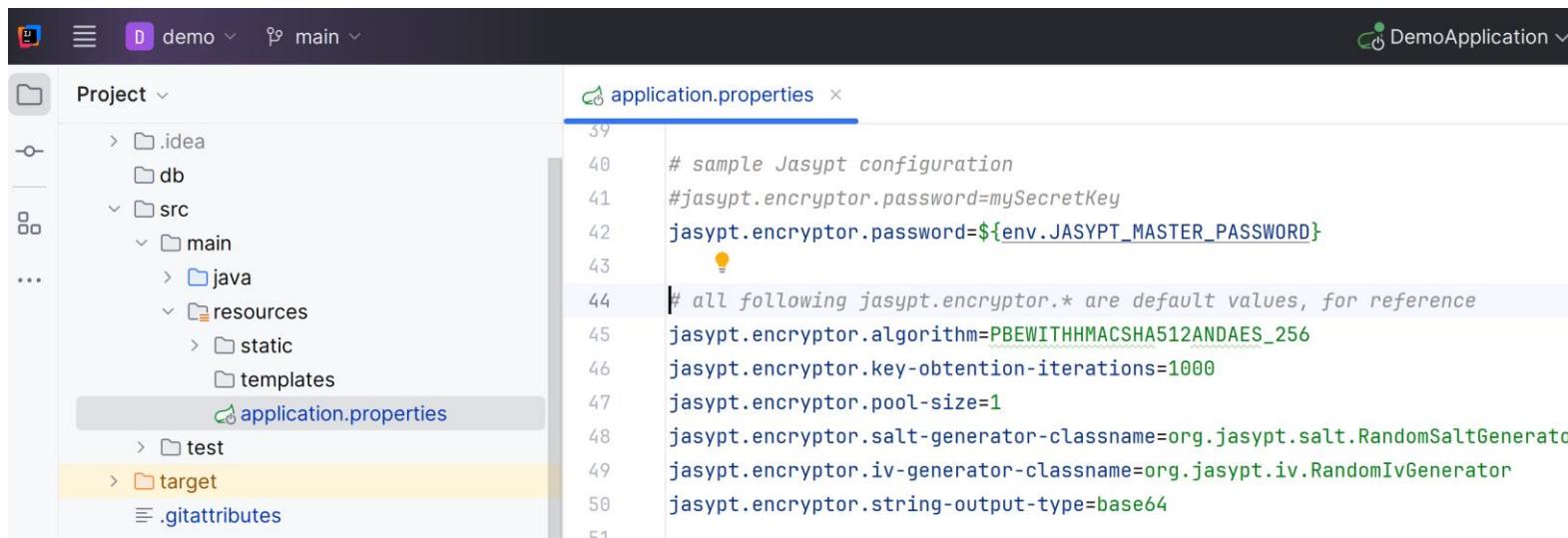
    public static void main(String[] args) {
        SpringApplication.run(DemoApplication.class, args);
    }
}
```

add

`@EnableEncryptableProperties`

to your app (on a the class main `@SpringBootApplication` or on any `@Configuration` class)

Jasypt [3/7] : default config of Jasypt Encryptor in application.properties



The screenshot shows the IntelliJ IDEA interface with the 'DemoApplication' project open. The 'application.properties' file is selected in the left-hand file tree. The code editor displays the following configuration:

```
# sample Jasypt configuration
#jasypt.encryptor.password=mySecretKey
jasypt.encryptor.password=${env.JASYPT_MASTER_PASSWORD}

# all following jasypt.encryptor.* are default values, for reference
jasypt.encryptor.algorithm=PBEWITHHMACSHA512ANDAES_256
jasypt.encryptor.key-obtention-iterations=1000
jasypt.encryptor.pool-size=1
jasypt.encryptor.salt-generator-classname=org.jasypt.salt.RandomSaltGenerator
jasypt.encryptor.iv-generator-classname=org.jasypt.iv.RandomIvGenerator
jasypt.encryptor.string-output-type=base64
```

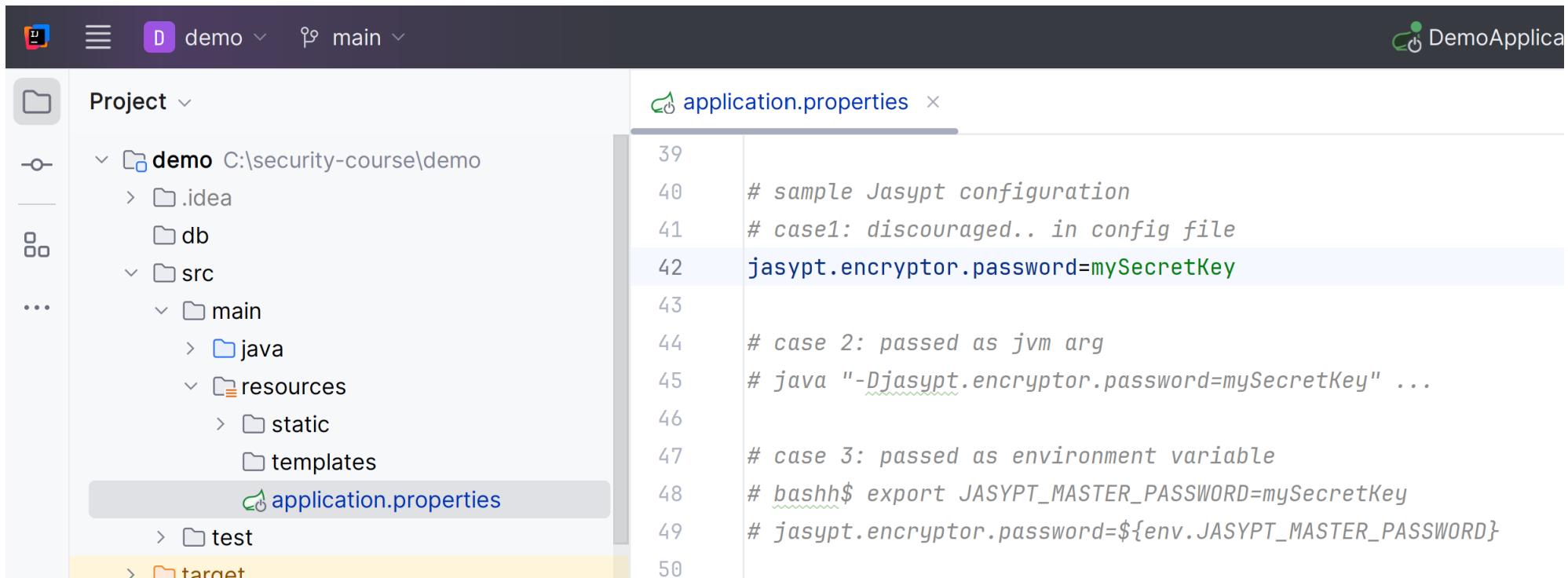
all following jasypt.encryptor.* are default values, for reference

```
jasypt.encryptor.algorithm=PBEWITHHMACSHA512ANDAES_256
jasypt.encryptor.key-obtention-iterations=1000
jasypt.encryptor.pool-size=1
jasypt.encryptor.salt-generator-classname=org.jasypt.salt.RandomSaltGenerator
jasypt.encryptor.iv-generator-classname=org.jasypt.iv.RandomIvGenerator
jasypt.encryptor.string-output-type=base64
```

edit file
src/main/resources/application.properties

Jasypt [4/7] case 1: simpler (but not recommended)

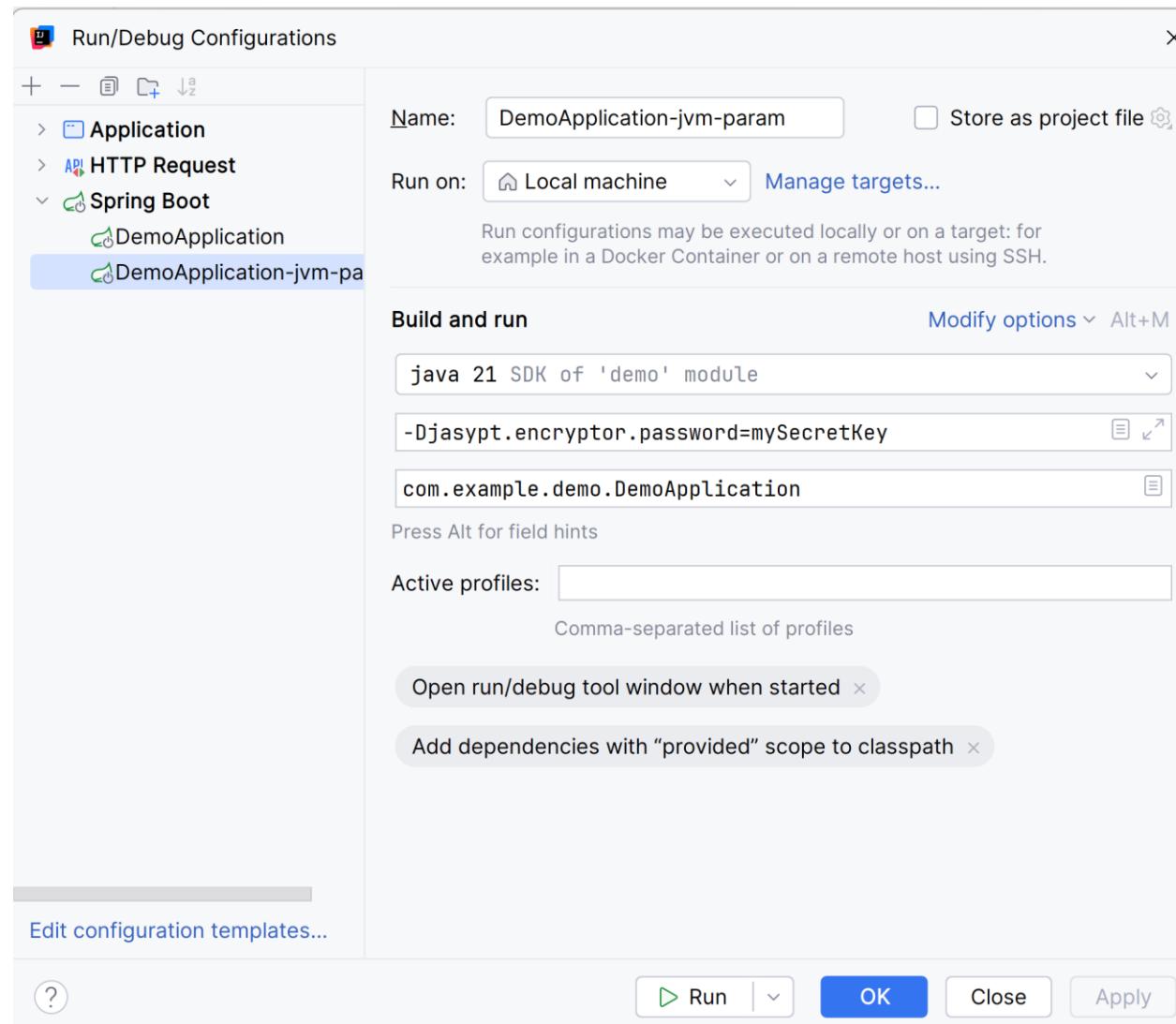
add jasypt encryptor password in configuration file



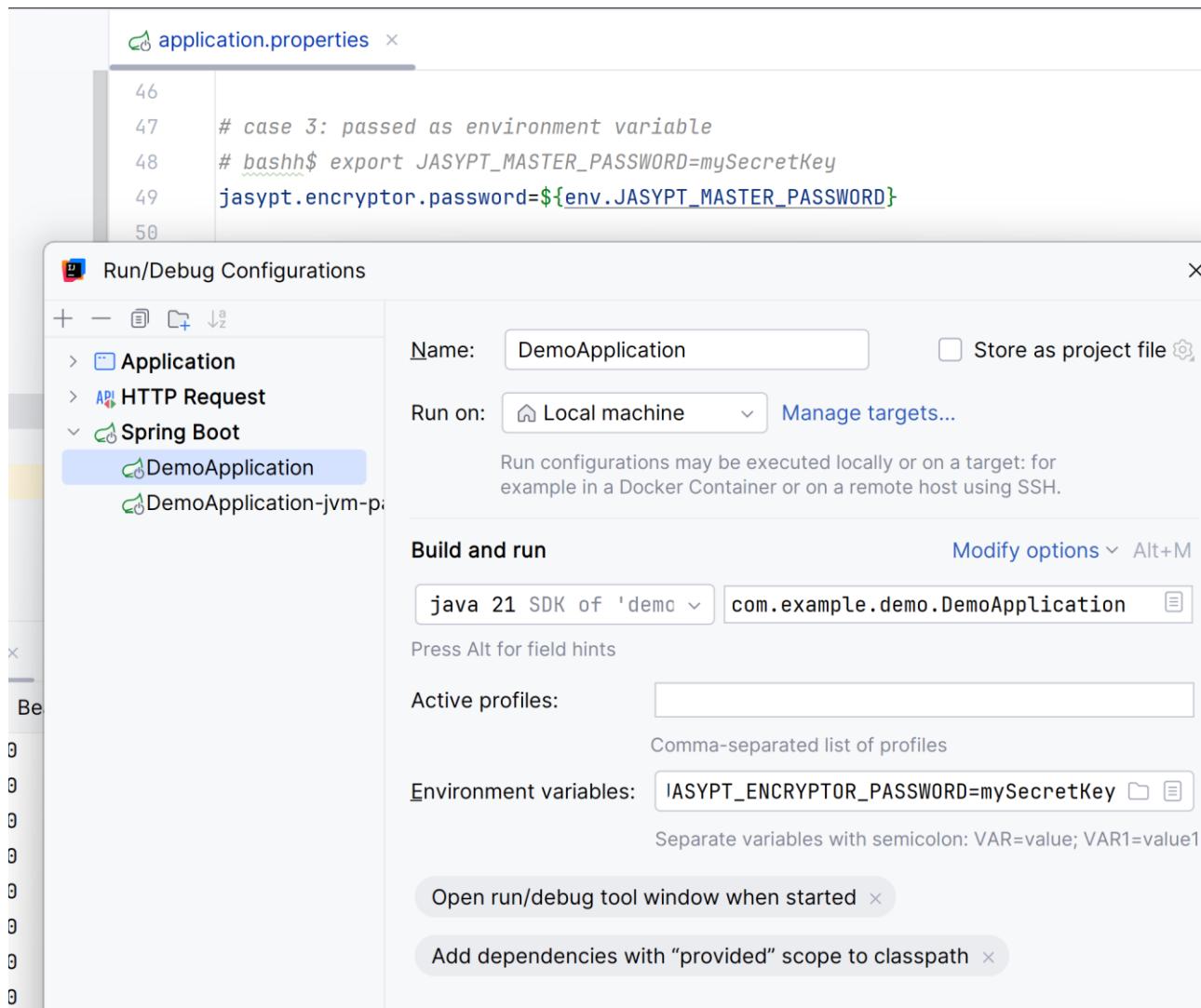
The screenshot shows a Java project structure in an IDE. The project is named 'demo' and contains a 'src' directory with a 'main' package. Inside 'main', there are 'java', 'resources', 'static', and 'templates' directories, along with an 'application.properties' file. The 'target' directory is at the bottom of the tree. On the right side, the 'application.properties' file is open in a code editor. The file content is as follows:

```
39 # sample Jasypt configuration
40 # case1: discouraged.. in config file
41 jasypt.encryptor.password=mySecretKey
42
43 # case 2: passed as jvm arg
44 # java "-Djasypt.encryptor.password=mySecretKey" ...
45
46 # case 3: passed as environment variable
47 # bashh$ export JASYPT_MASTER_PASSWORD=mySecretKey
48 # jasypt.encryptor.password=${env.JASYPT_MASTER_PASSWORD}
49
50
```

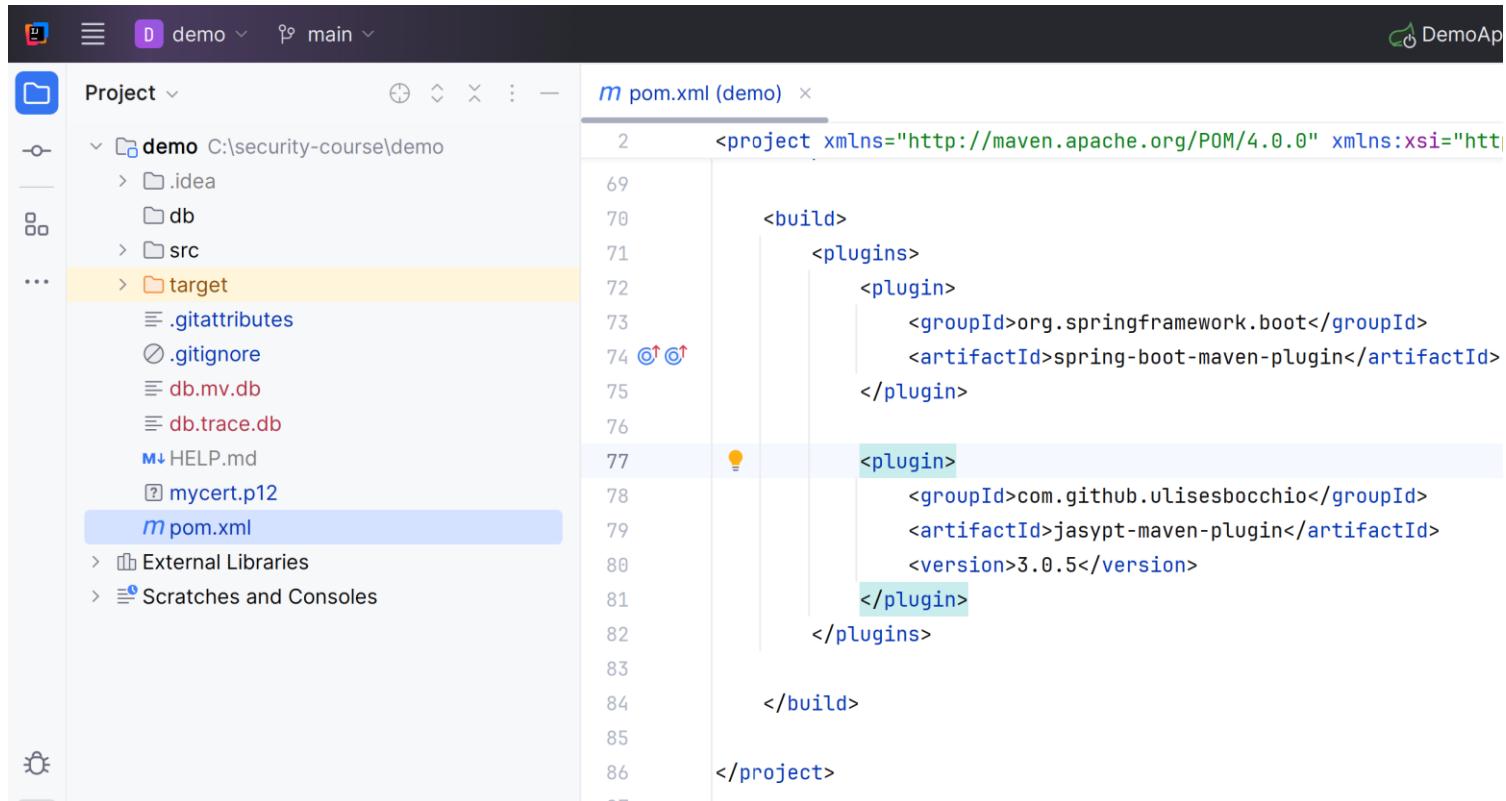
(Optionnal) Jasypt [4/7] case 2: with JVM param -Djasypt.encryptor.password=...



(Optionnal) Jasypt [4/7] case 3: with ENVIRONMENT variable + conf



Jasypt [6/7] : compute the encryption of a secret value



The screenshot shows the IntelliJ IDEA interface with a Maven project named 'demo'. The 'target' directory is selected and highlighted in yellow. The 'pom.xml' file is open in the editor, displaying the following XML code:

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="htt|  
  ...  
<build>  
  <plugins>  
    <plugin>  
      <groupId>org.springframework.boot</groupId>  
      <artifactId>spring-boot-maven-plugin</artifactId>  
    </plugin>  
    <plugin>  
      <groupId>com.github.ulisesbocchio</groupId>  
      <artifactId>jasypt-maven-plugin</artifactId>  
      <version>3.0.5</version>  
    </plugin>  
  </plugins>  
</build>  
</project>
```

Could use directly "java -jar jasypt.jar .."
but would need to download manually
=> Simpler to use a maven plugin,
edit file pom.xml <build><plugins> ...

```
<plugin>  
  <groupId>com.github.ulisesbocchio</groupId>  
  <artifactId>jasypt-maven-plugin</artifactId>  
  <version>3.0.5</version>  
</plugin>
```

Jasypt [6/7] : compute the encryption of a secret value

open windows 'cmd' ... need to have maven downloaded and in %PATH% environment variable

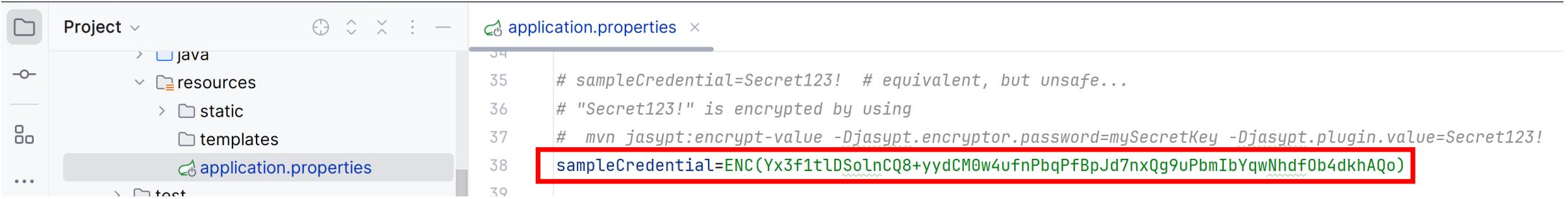
mvn jasypt:encrypt-value -Djasypt.encryptor.password=mySecretKey -Djasypt.plugin.value=Secret123!

```
Cmder
arnaud@DesktopArnaud /cygdrive/c/security-course/demo
$ mvn jasypt:encrypt-value -Djasypt.encryptor.password=mySecretKey -Djasypt.plugin.value=Secret123!
[INFO] Scanning for projects...
[INFO]
[INFO] -----< com.example:demo >-----
[INFO] Building demo 0.0.1-SNAPSHOT
[INFO]   from pom.xml
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- jasypt:3.0.5:encrypt-value (default-cli) @ demo ---
[INFO] Starting MavenCli v3.9.9 using Java 21.0.6 on DesktopArnaud with PID 436 (C:\apps\maven\mvn-lat
:\security-course\demo)
[INFO] No active profile set, falling back to 1 default profile: "default"
[INFO]
```

Jasypt [6/7] : compute the encryption of a secret value

```
[INFO] Started MavenCI1 in 1.201 seconds (JVM running for 3.729)
[INFO] Active Profiles: Default
[INFO] Encrypting value Secret123!
[INFO] String Encryptor custom Bean not found with name 'jasyptStringEncryptor'. Initializing Default String Encryptor
[INFO] Encryptor config not found for property jasypt.encryptor.algorithm, using default value: PBEWITHHMACSHA512ANDAES_256
[INFO] Encryptor config not found for property jasypt.encryptor.key-obtention-iterations, using default value: 1000
[INFO] Encryptor config not found for property jasypt.encryptor.pool-size, using default value: 1
[INFO] Encryptor config not found for property jasypt.encryptor.provider-name, using default value: null
[INFO] Encryptor config not found for property jasypt.encryptor.provider-class-name, using default value: null
[INFO] Encryptor config not found for property jasypt.encryptor.salt-generator-classname, using default value: org.jasypt.salt.RandomSaltGenerator
[INFO] Encryptor config not found for property jasypt.encryptor.iv-generator-classname, using default value: org.jasypt.iv.RandomIvGenerator
[INFO] Encryptor config not found for property jasypt.encryptor.string-output-type, using default value: base64
[INFO]
[INFO] -----  
ENC(Yx3f1t1DS0lnCQ8+yydCM0w4ufnPbqPfBpJd7nxQg9uPbmIbYqwNhdf0b4dkhAQo)
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 2.427 s
[INFO] Finished at: 2025-03-07T21:13:19+01:00
[INFO] -----
```

Jasypt [6/7] : add encrypted value in application.properties



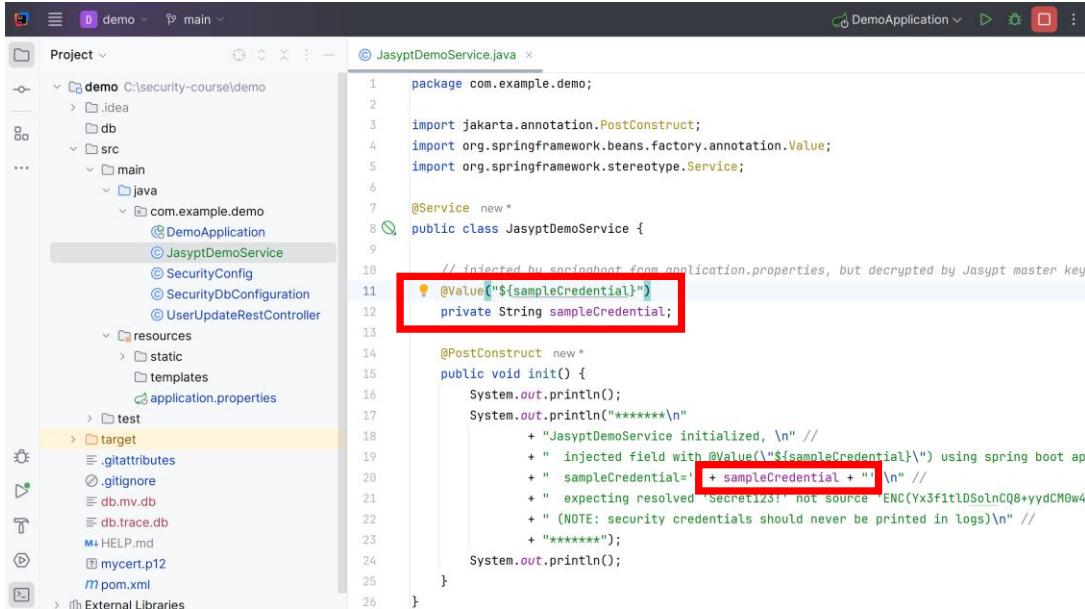
```
# sampleCredential=Secret123! # equivalent, but unsafe...
# "Secret123!" is encrypted by using
# mvn jasypt:encrypt-value -Djasypt.encryptor.password=mySecretKey -Djasypt.plugin.value=Secret123!
sampleCredential=ENC(Yx3f1tIDSolnCQ8+yydCM0w4ufnPbqPfBpJd7nxQg9uPbmIbYqwNhdfOb4dkhAQo)
```

edit src/main/resources/application.properties

```
sampleCredential=ENC(Yx3f1tIDSolnCQ8+yydCM0w4ufnPbqPfBpJd7nxQg9uPbmIbYqwNhdfOb4dkhAQo)
```

NOTICE: because of random Salting, every re-generation will get a different value

Jasypt [7/7] :use injected value in java code



```
1 package com.example.demo;
2
3 import jakarta.annotation.PostConstruct;
4 import org.springframework.beans.factory.annotation.Value;
5 import org.springframework.stereotype.Service;
6
7 @Service
8 public class JasyptDemoService {
9
10    // injected by springboot from application.properties, but decrypted by Jasypt master key
11    @Value("${sampleCredential}")
12    private String sampleCredential;
13
14    @PostConstruct
15    public void init() {
16        System.out.println();
17        System.out.println("*****\n" +
18            + "JasyptDemoService initialized, \n" //
19            + " injected field with @Value(\"${sampleCredential}\") using spring boot app" +
20            + " sampleCredential=" + sampleCredential + "\n" //
21            + " expecting resolved 'Secret123!' not source 'ENC(Yx3f1tDSolnCQ8+yydCM0w4u" +
22            + "(NOTE: security credentials should never be printed in logs)\n" //
23            + "*****");
24        System.out.println();
25    }
26}
```

add new java class JasyptDemoService

```
package com.example.demo;
import jakarta.annotation.PostConstruct;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.stereotype.Service;

@Service
public class JasyptDemoService {

    // injected by springboot from application.properties, but decrypted by Jasypt master key
    @Value("${sampleCredential}")
    private String sampleCredential;

    @PostConstruct
    public void init() {
        System.out.println();
        System.out.println("*****\n" +
            + "JasyptDemoService initialized, \n" //
            + " injected field with @Value(\"${sampleCredential}\") using spring boot application.properties \n" //
            + " sampleCredential=" + sampleCredential + "\n" //
            + " expecting resolved 'Secret123!' not source
            'ENC(Yx3f1tDSolnCQ8+yydCM0w4ufnPbqPfBpJd7nxQg9uPbmIbYqwNhdfOb4dkhAQo)' \n" //
            + " (NOTE: security credentials should never be printed in logs)\n" //
            + "*****");
        System.out.println();
    }
}
```

Jasypt [7/7]: Run see startup logs

The screenshot shows a Java IDE interface with the following components:

- Project Explorer:** Shows the package structure `com.example.demo` containing files: `DemoApplication`, `JasyptDemoService`, `SecurityConfig`, `SecurityDbConfiguration`, and `UserUpdateRestController`. `DemoApplication` is selected.
- Code Editor:** Displays the `DemoApplication.java` code. A red box highlights the line `11 > public static void main(String[] args) { SpringApplication.run(DemoAppli`.
- Debug Tool Window:** Titled "Debug DemoApplication". It includes a toolbar with icons for play, pause, step, and stop. Below the toolbar is a log pane showing application startup messages. A red box highlights the word "Secret123!" in the log output.
- Log Output:** Shows the following log entries:
 - 2025-03-07T21:30:42.781+01:00 INFO 27372 --- [demo] [main] c.u.j.c.StringEncryptorBuilder : Encrypt
 - 2025-03-07T21:30:42.781+01:00 INFO 27372 --- [demo] [main] c.u.j.c.StringEncryptorBuilder : Encrypt
 - *****
JasyptDemoService initialized,
 injected field with @Value("\${sampleCredential}") using spring boot application.properties
 sampleCredential: 'Secret123!' **Red box highlights this line**
 expecting resolved 'Secret123!' not source 'ENC(Yx3f1t1DSolnCQ8+yydCM0w4ufnPbqPfBpJd7nxQg9uPbmIbYqwNhf0b4dkhAQo)'
 (NOTE: security credentials should never be printed in logs)

 - 2025-03-07T21:51:50.440+01:00 WARN 27372 --- [demo] [l-1 housekeeper] com.zaxxer.hikari.pool.HikariPool : HikariP
 - Disconnected from the target VM, address: '127.0.0.1:51784', transport: 'socket'
 - 2025-03-07T21:51:50.525+01:00 WARN 27372 --- [demo] [main] ConfigServletWebServerApplicationContext : Excepti

Jasypt [7/7]: Debug with Breakpoint (debug as you should never print credential in logs!)

The screenshot shows an IDE interface with the following details:

- Project View:** Shows the project structure under "src/main/java".
 - application.properties** is selected.
 - target** is highlighted.
- Code Editor:** The file **JasyptDemoService.java** is open, showing Java code. A red box highlights the **System.out.println()** call at line 25.
- Debug View:** The "Debug" tab is active, showing the process **DemoApplication** is running. The stack trace shows the method **init** being executed.
- Breakpoint:** A red box highlights the breakpoint icon on the left margin of the code editor at line 25.
- Watch View:** The expression **sampleCredential = "Secret123!"** is shown in the watch list, also highlighted with a red box.
- Status Bar:** Shows the path **demo > src > main > java > com > example > demo > JasyptDemoService > init**, and the status **Supermaven Free Tier 24:1 CRLF UTF-8 4 spaces**.

Using Hashicorp Vault

Download + Unzip Hashicorp Vault

<https://developer.hashicorp.com/vault/install>

The screenshot shows the Hashicorp Vault installation page on developer.hashicorp.com. The URL in the browser bar is <https://developer.hashicorp.com/vault/install>. The page title is "Install Vault". On the left sidebar, under "Operating Systems", "macOS" is selected. The main content area features a yellow "Install Vault" button and a "macOS" section with package manager instructions:

```
brew tap hashicorp/tap
brew install hashicorp/tap/vault
```

Below this is a "Binary download" section. To the right, there's an "About Vault" summary and links to "Featured docs" like "What is Vault?", "Use Cases", and "Developer Quick Start". A blue banner at the bottom right promotes "HCP Vault Dedicated".

Run in "-dev" mode

```
C:\security-course\vault> vault.exe server -dev
```

Cmder

```
arnaud@DesktopArnaud /cygdrive/c/security-course/vault
$ ./vault.exe server -dev
==> Vault server configuration:
```

Administrative Namespace:

 Api Address: http://127.0.0.1:8200

 Cgo: disabled

 Cluster Address: https://127.0.0.1:8201

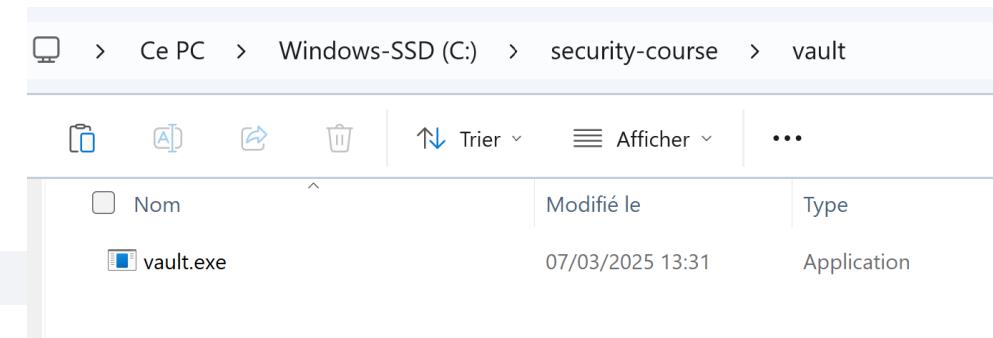
Environment Variables: , ALLUSERSPROFILE, ANSICON, ANSICON_DEF, APPDATA, CMDER_ROOT, COMMONPROGRAMW6432, CONEMUANSI, CONEMUARGS, CONEMUBACKHWND, CONEMUBASEDIR, CONEMUBASEDIRSHVE, CONEMUHWND, CONEMUHOOKS, CONEMUPID, CONEMUPALETTE, CONEMUSERVPID, CONEMUTASK, CONEMWSER_APP_PROFILE_STRING, FPS_BROWSER_USER_PROFILE_STRING, GRADLE_HOME, HADOOP_HOME, HOME, ATA, LOGONSERVER, M2_HOME, MAVEN_OPTS, NODE_HOME, NODE_OPTIONS, NUMBER_OF_PROCESSORS, OLDARCHITECTURE, PROCESSOR_ARCHITEW6432, PROCESSOR_IDENTIFIER, PROCESSOR_LEVEL, PROCESSOR_RELYSPARK PYTHON, ProgramData, ProgramFiles(x86), ProgramW6432, SBT_HOME, SESSIONNAME, SHELLSER, USERDOMAIN, USERDOMAIN_ROAMINGPROFILE, USERNAME, USERPROFILE, VBOX_MSI_INSTALL_PATH,

 Go Version: go1.23.6

 Listener 1: tcp (addr: "127.0.0.1:8200", cluster address: "127.0.0.1:8201",
x_request_size: "33554432", tls: "disabled")

 Log Level:

 Mlock: supported: false, enabled: false



See console log root "token"

(NOTICE: "-dev" = in-memory only, no secrets saved... and token regenerated at startup)

The screenshot shows a terminal window titled 'Cmder' with the following content:

```
2025-03-07T13:32:55.363+0100 [INFO] core: restoring leases
2025-03-07T13:32:55.363+0100 [INFO] expiration: lease restore complete
2025-03-07T13:32:55.363+0100 [INFO] identity: entities restored
2025-03-07T13:32:55.363+0100 [INFO] identity: groups restored
2025-03-07T13:32:55.363+0100 [INFO] core: post-unseal setup complete
2025-03-07T13:32:55.363+0100 [INFO] core: vault is unsealed
2025-03-07T13:32:55.363+0100 [INFO] core: successful mount: namespace="" path=secret/ type=kv version=
WARNING! dev mode is enabled! In this mode, Vault runs entirely in-memory
and starts unsealed with a single unseal key. The root token is already
authenticated to the CLI, so you can immediately begin using Vault.

You may need to set the following environment variables:

PowerShell:
$env:VAULT_ADDR="http://127.0.0.1:8200"
cmd.exe:
set VAULT_ADDR=http://127.0.0.1:8200

The unseal key and root token are displayed below in case you want to
seal/unseal the Vault or re-authenticate.

Unseal Key: FGVqgCKNAsb9z0ep9lvvfvzDxZ2C2/yW04aGqAzWxbk=
Root Token: hvs.flDs6tHy0URvX3DF1qUsJu1k

Development mode should NOT be used in production installations!
```

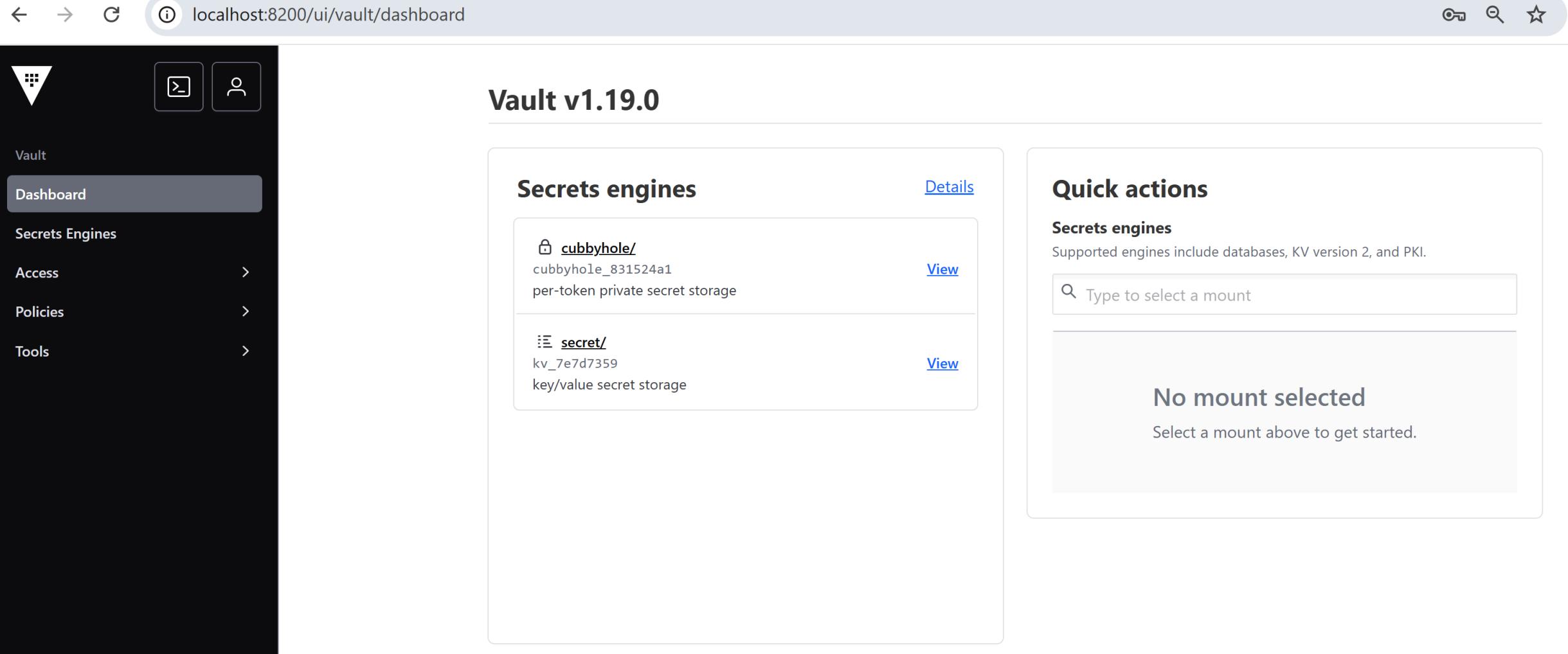
Open Vault Http UI

<http://localhost:8200>

The screenshot shows a web browser window with the URL `localhost:8200/ui/vault/auth?with=token` in the address bar. The main content is a 'Sign in to Vault' form. At the top is a dark gray header with a small circular icon containing a question mark. Below the header is a large, dark gray downward-pointing triangle icon. The form itself has a light gray background. It features a title 'Sign in to Vault' in bold black font. Below the title is a 'Method' section with a dropdown menu set to 'Token'. Underneath is a 'Token' section with an input field. At the bottom of the form is a blue 'Sign in' button. The entire form is contained within a white rectangular box.

Contact your administrator for login credentials.

Login... UI Home Page



The screenshot shows the HashiCorp Vault UI Home Page. At the top, there is a navigation bar with icons for back, forward, and refresh, followed by the URL "localhost:8200/ui/vault/dashboard". On the far right of the navigation bar are three small icons: a key, a magnifying glass, and a star.

The main content area has a dark header with the Vault logo and two small icons. Below the header is a title "Vault v1.19.0". To the left is a sidebar with the following menu items:

- Vault
- Dashboard** (highlighted)
- Secrets Engines
- Access >
- Policies >
- Tools >

The main content area contains two main sections:

- Secrets engines**: This section lists two engines:
 - cubbyhole/**: cubbyhole_831524a1, per-token private secret storage. It includes a "View" link and a "Details" link.
 - secret/**: kv_7e7d7359, key/value secret storage. It includes a "View" link.
- Quick actions**: This section includes a heading "Secrets engines" and a note that supported engines include databases, KV version 2, and PKI. It features a search bar with the placeholder "Type to select a mount".

At the bottom right of the main content area, there is a message: "No mount selected" and "Select a mount above to get started."

Create a Secret

The screenshot shows the HashiCorp Vault UI interface. On the left is a dark sidebar with navigation links: Vault, Dashboard, Secrets Engines (which is selected and highlighted in grey), Access, Policies, and Tools. At the top right of the main content area are icons for key, search, and star. The URL in the browser bar is `localhost:8200/ui/vault/secrets/secret/kv/list`.

The main content area has a header "Create Secret". Below it is a "JSON" toggle switch. The "Path for this secret" section contains the path "my-app". The "Secret data" section contains two fields: "secret1" and a redacted value represented by five dots. There is also a "Show secret metadata" link. At the bottom are "Save" and "Cancel" buttons.

Path for this secret
Names with forward slashes define hierarchical path structures.
my-app

Secret data
secret1 Add

Show secret metadata

Save Cancel

(Optionnal) Creating secret with cmd

./vault kv put secret/application1 "myVaultSecret=Secret123"

```
$ ./vault kv put secret/application1 "myVaultSecret=Secret123"
===== Secret Path =====
secret/data/application1

===== Metadata =====
Key          Value
---          ---
created_time 2025-03-07T23:04:16.9254565Z
custom_metadata <nil>
deletion_time n/a
destroyed      false
version        3
```

Configured Secret in Vault

path=application1

secretName=myVaultSecret (secretValue=Secret123)

The screenshot shows the HashiCorp Vault UI interface. On the left, a dark sidebar menu includes 'Vault', 'Dashboard', 'Secrets Engines' (which is highlighted), 'Access', 'Policies', 'Tools', 'Monitoring', 'Client Count', and 'Seal Vault'. The main content area has a light background. At the top, a header bar shows the URL '127.0.0.1:8200/ui/vault/secrets/secret/kv/application1/details?version=1'. Below the header, the breadcrumb navigation shows 'Secrets / secret / application1'. The title 'application1' is displayed above a tab bar with 'Overview', 'Secret' (which is selected and highlighted in blue), 'Metadata', 'Paths', and 'Version History'. Underneath the tabs, there is a 'JSON' toggle switch and buttons for 'Delete' and 'Destroy'. A table-like structure displays a single secret entry:

Key	Value
myVaultSecret	[REDACTED]

Use springboot cloud + Hashicorp Vault
to resolve secrets in config

add pom.xml dependency spring-cloud-starter-vault-config

m pom.xml (demo) ×

```
2   <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w
32     <dependencies> ⚡ Add Starters...
55
56       <dependency>
57         <groupId>org.springframework.cloud</groupId>
58         <artifactId>spring-cloud-starter-vault-config</artifactId>
59         <version>4.1.2</version>
60       </dependency>
61
```

edit pom.xml, add <dependencies>...

```
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-vault-config</artifactId>
  <version>4.1.2</version>
</dependency>
```

Configure springboot - Vault

application.properties ×

```
62  
63 # Spring Cloud HashiCorp Vault configuration  
64  
65 #spring.cloud.vault.enabled=true  
66 spring.cloud.vault.uri=http://localhost:8200  
67 spring.cloud.vault.token=hvs.6qzL7qdKN2EjtbieAzhXqqTZ  
68 spring.cloud.vault.kv.backend=secret  
69 spring.cloud.vault.kv.default-context=application1  
70  
71 spring.config.import=vault://
```

edit file
src/main/resources/
application.properties

```
# Spring Cloud HashiCorp Vault configuration  
spring.cloud.vault.uri=http://localhost:8200  
spring.cloud.vault.token=*****your token  
spring.cloud.vault.kv.backend=secret  
spring.cloud.vault.kv.default-context=application1  
  
spring.config.import=vault://  
  
# myVaultSecret=.. cf overriden injected from Vault 'application1/secret/myVaultSecret'
```

Add Java code to use injected value

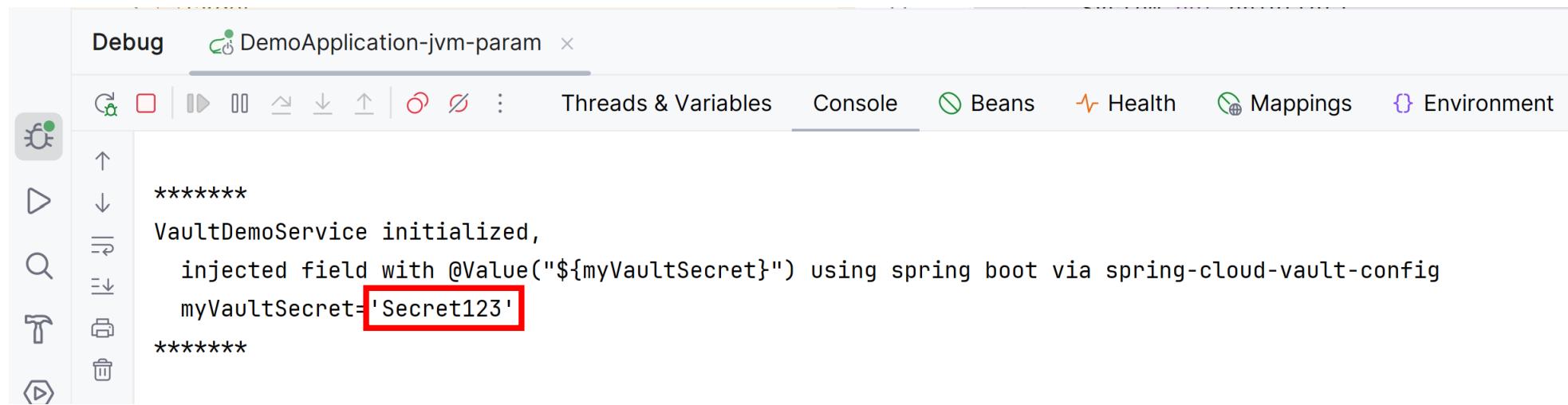
edit new java class
VaultDemoService

```
@Service
public class VaultDemoService {

    // injected by springboot from Vault, via spring-cloud-vault-
    config
    @Value("${myVaultSecret}")
    private String myVaultSecret;

    @PostConstruct
    public void init() {
        System.out.println();
        System.out.println("*****\n"
            + "VaultDemoService initialized, \n" //
            + " injected field with @Value(\"${myVaultSecret}\")"
            using spring boot via spring-cloud-vault-config \n" //
            + " myVaultSecret=\"" + myVaultSecret + "\" \n" //
            + "*****");
        System.out.println();
    }
}
```

Run/Debug



The screenshot shows the IntelliJ IDEA interface with the "Debug" tool window open. The title bar indicates the session is named "DemoApplication-jvm-param". The tool window has several tabs: "Threads & Variables", "Console", "Beans", "Health", "Mappings", and "Environment". The "Console" tab is selected, displaying application logs. The logs show the initialization of the "VaultDemoService" and the injection of a field with the value "Secret123". The line containing "myVaultSecret='Secret123'" is highlighted with a red box.

```
*****  
VaultDemoService initialized,  
 injected field with @Value("${myVaultSecret}") using spring boot via spring-cloud-vault-config  
 myVaultSecret='Secret123'  
*****
```

NOTICE... if property (Secret) not found => Error at startup

The screenshot shows an IDE interface with the following components:

- Code Editor:** Displays a Java file named `VaultDemoService.java`. A specific line of code is highlighted with a red box:

```
10     // injected by springboot from Vault, via spring-cloud-vault-config
11     @Value("${myVaultSecret_INVALID}")
12     private String myVaultSecret;
```
- File Explorer:** Shows the project structure with files like `_old_bootstrap.yml` and `application.properties`.
- Debug Console:** Shows the stack trace of an error. The error message is highlighted with a red box:

```
Caused by: org.springframework.util.PlaceholderResolutionException Create breakpoint Could not resolve placeholder 'myVaultSecret_INVALID' in value "${myVaultSecret_INV..."
```
- Toolbars and Side Panels:** Various icons for navigation, search, and configuration are visible on the left side.

Questions ?