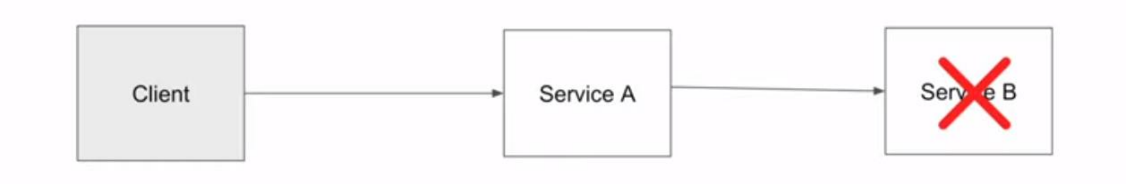
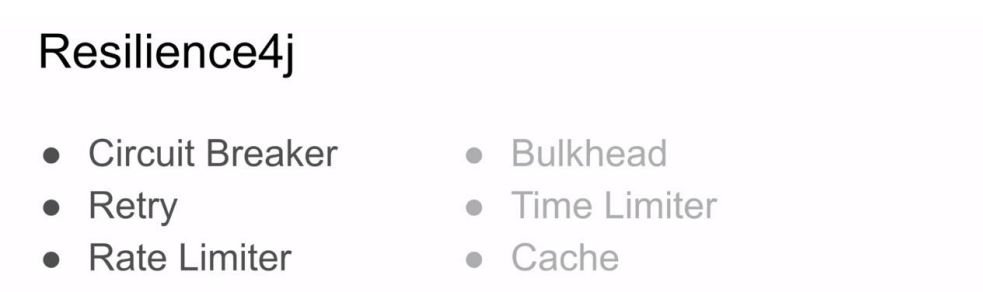
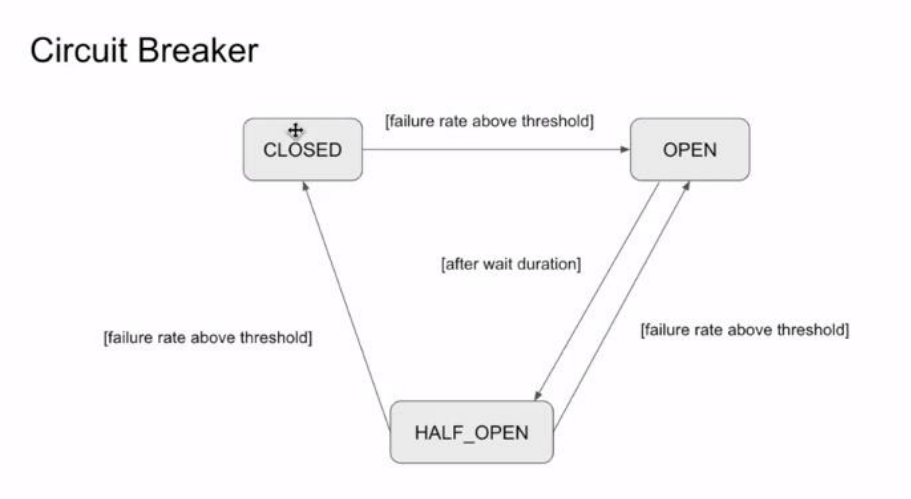
Resilience4j Tutorial with Spring Boot | Circuit Breaker, Retry, Rate Limiter

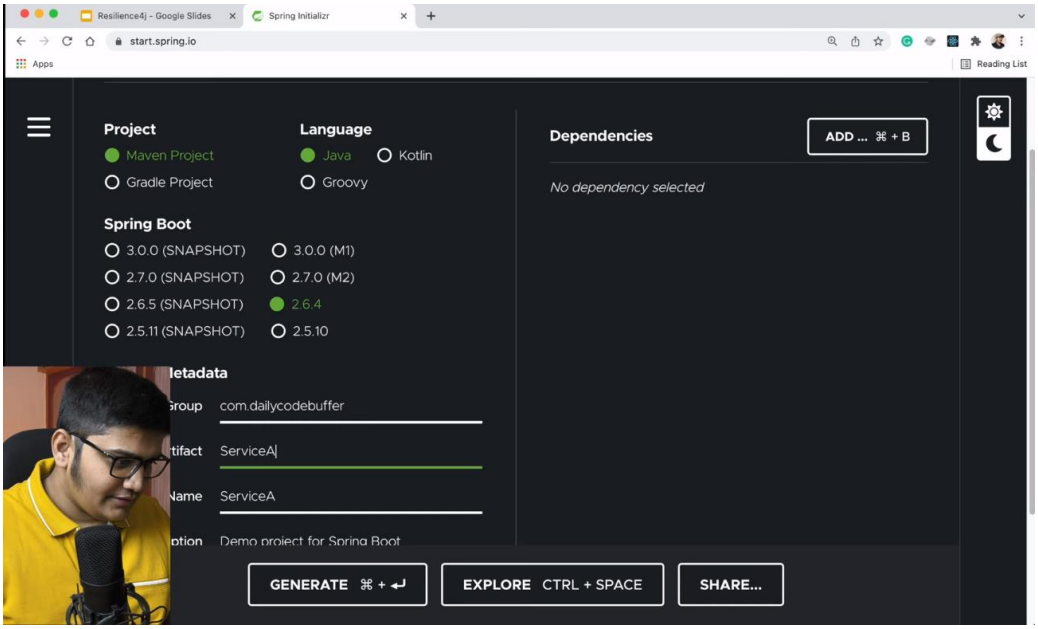


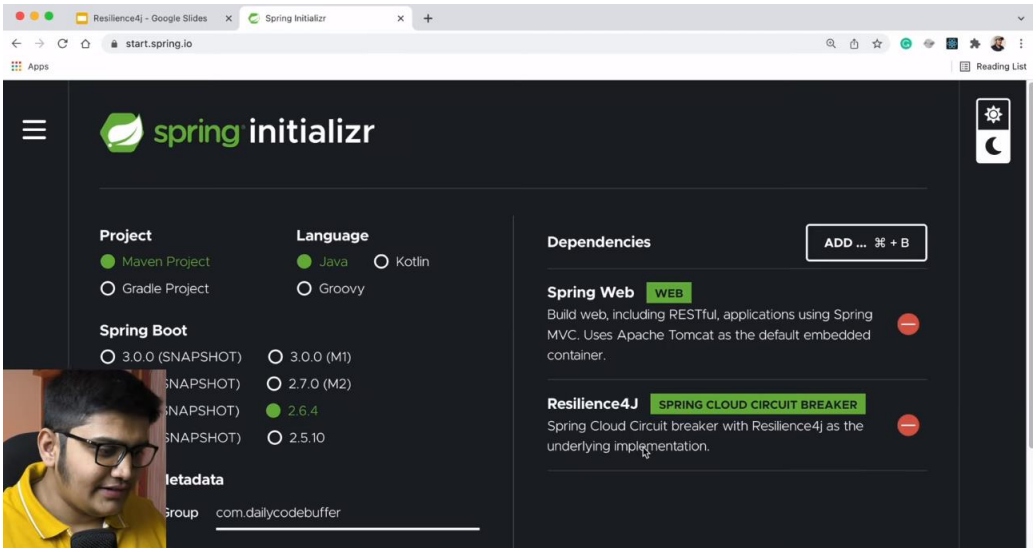


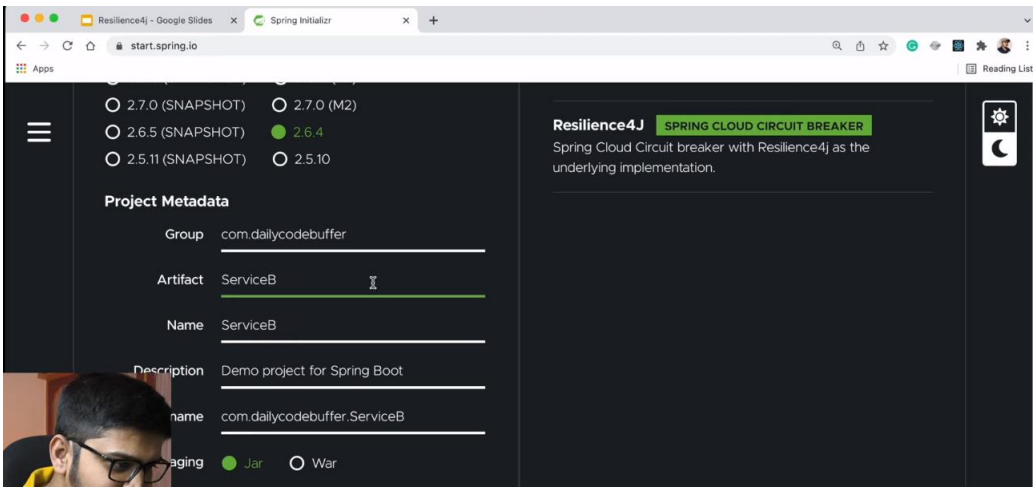
There are different modules that we can use within our architecture

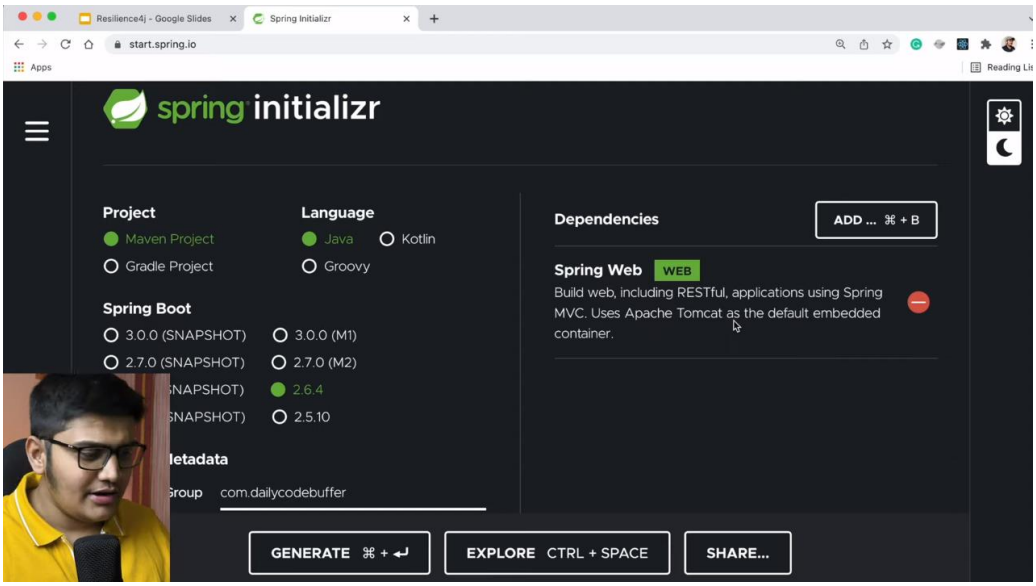


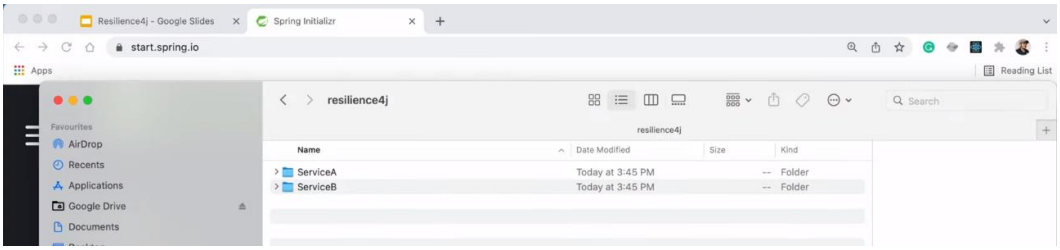
You can define the request X% **threshold** per N requests for the *circuit* to **open** for the T **wait period** in seconds. We need to add the configuration for this in our YAML file.

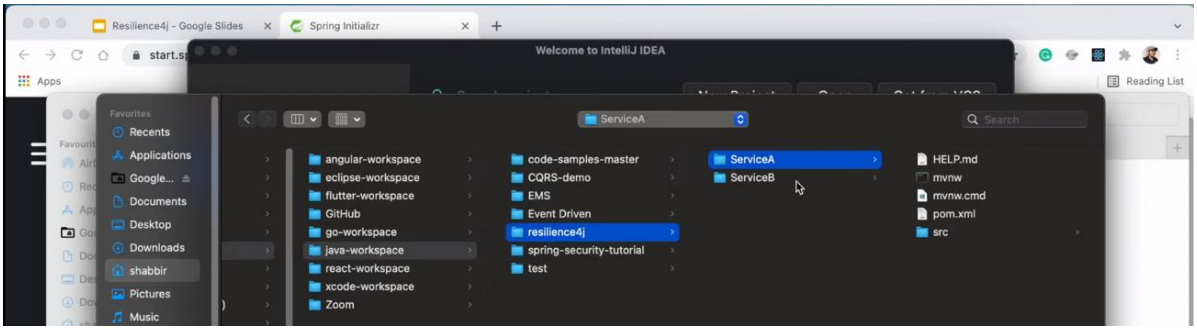


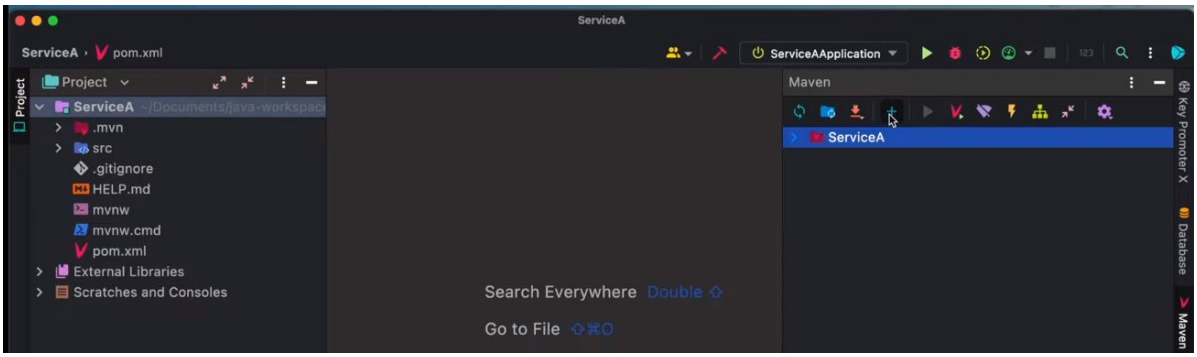

Resilience4j - Google Slides
Spring Initializr
start.spring.io
Apps
Reading List
spring initializr
Project
Maven Project
Gradle Project
Language
Java
Kotlin
Groovy
Spring Boot
3.0.0 (SNAPSHOT)
3.0.0 (M1)
2.7.0 (M2)
2.6.4
2.5.10
Dependencies
ADD ... ⌘ + B
Spring Web
WEB
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.
Resilience4J
SPRING CLOUD CIRCUIT BREAKER
Spring Cloud Circuit breaker with Resilience4j as the underlying implementation.
com.dailycodebuffer

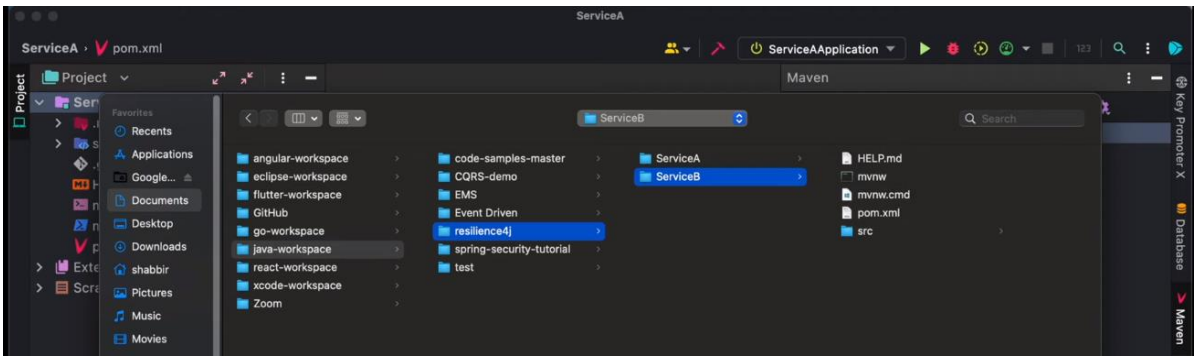

Resilience4j - Google Slides
Spring Initializr
start.spring.io
Apps
Reading List
2.7.0 (SNAPSHOT)
2.7.0 (M2)
2.6.5 (SNAPSHOT)
2.6.4
2.5.11 (SNAPSHOT)
2.5.10
Resilience4J
SPRING CLOUD CIRCUIT BREAKER
Spring Cloud Circuit breaker with Resilience4j as the underlying implementation.
Project Metadata
Group
com.dailycodebuffer
Artifact
ServiceB
Name
ServiceB
Description
Demo project for Spring Boot
com.dailycodebuffer.ServiceB
Jar
War

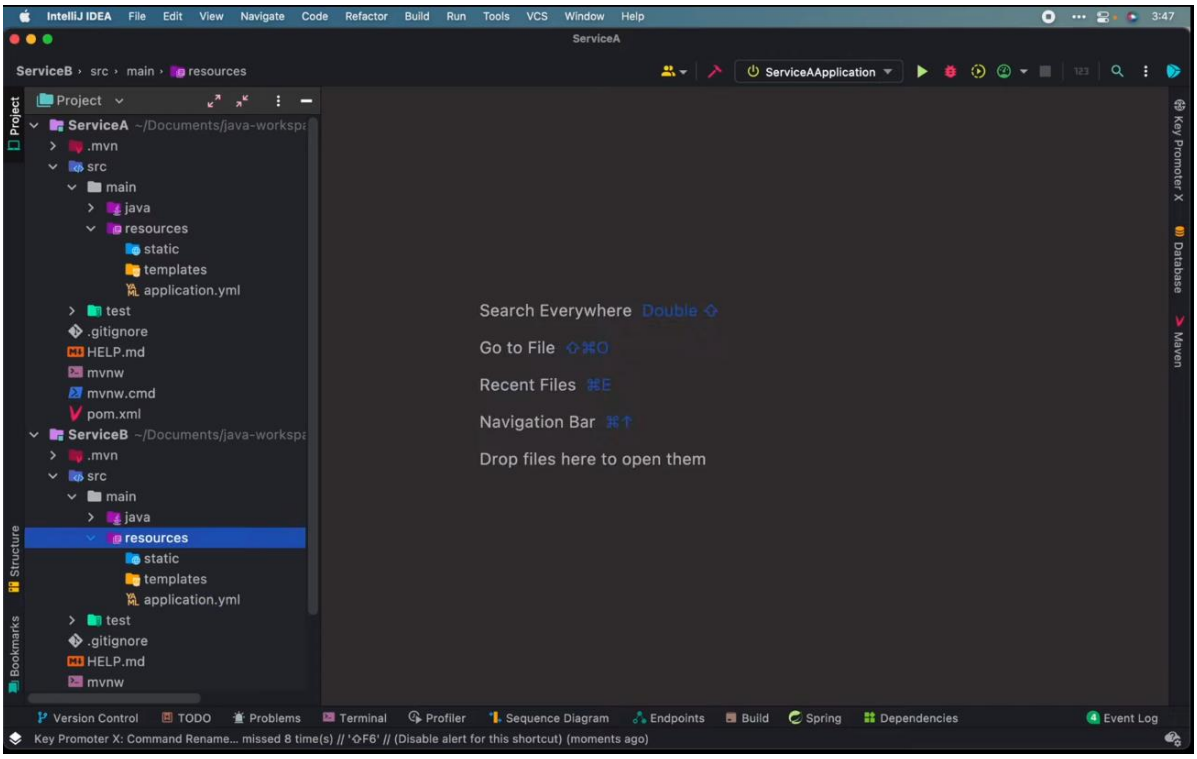

Resilience4j - Google Slides
Spring Initializr
start.spring.io
Apps
spring initializr
Project
Maven Project
Gradle Project
Language
Java
Kotlin
Groovy
Spring Boot
3.0.0 (SNAPSHOT)
3.0.0 (M1)
2.7.0 (SNAPSHOT)
2.7.0 (M2)
2.6.4
2.5.10
Dependencies
ADD ... ⌘ + B
Spring Web
WEB
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.
com.dailycodebuffer
GENERATE ⌘ + ↵
EXPLORE CTRL + SPACE
SHARE...

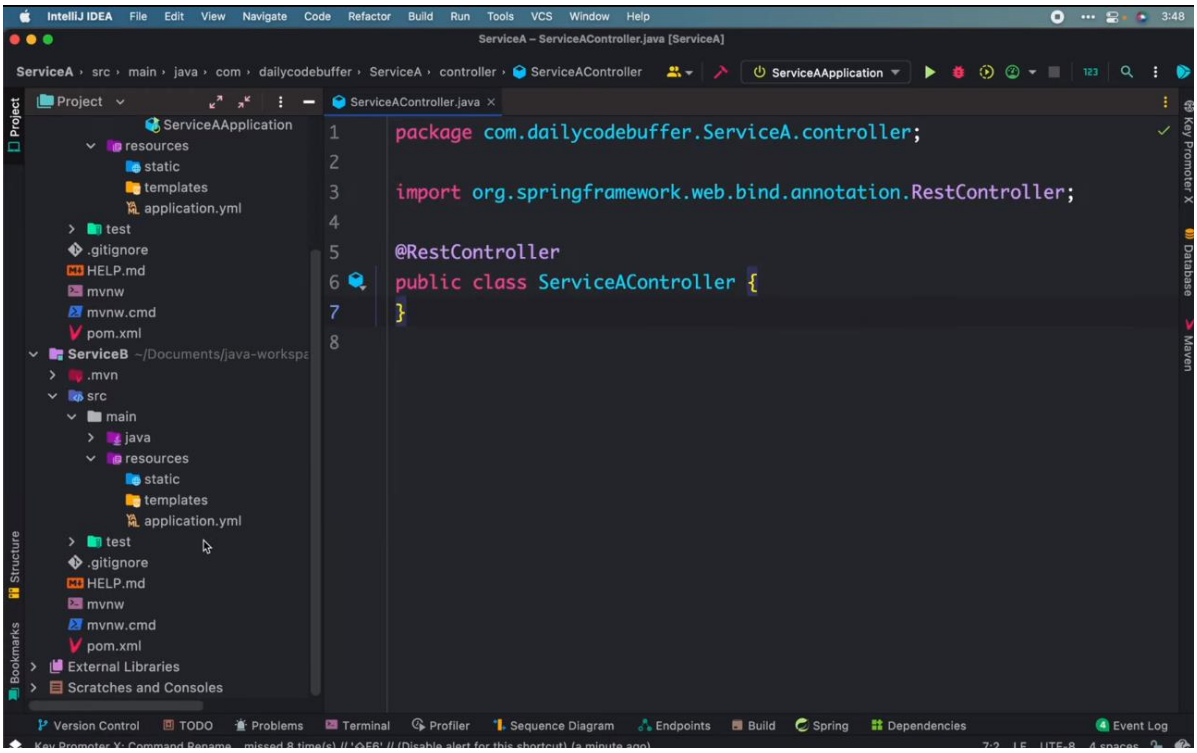

Resilience4j - Google Slides
Spring Initializr
start.spring.io
Apps
Reading List
resilience4j
Favourites
AirDrop
Recents
Applications
Google Drive
Documents
Name
Date Modified
Size
Kind
ServiceA
Today at 3:45 PM
Folder
ServiceB
Today at 3:45 PM
Folder

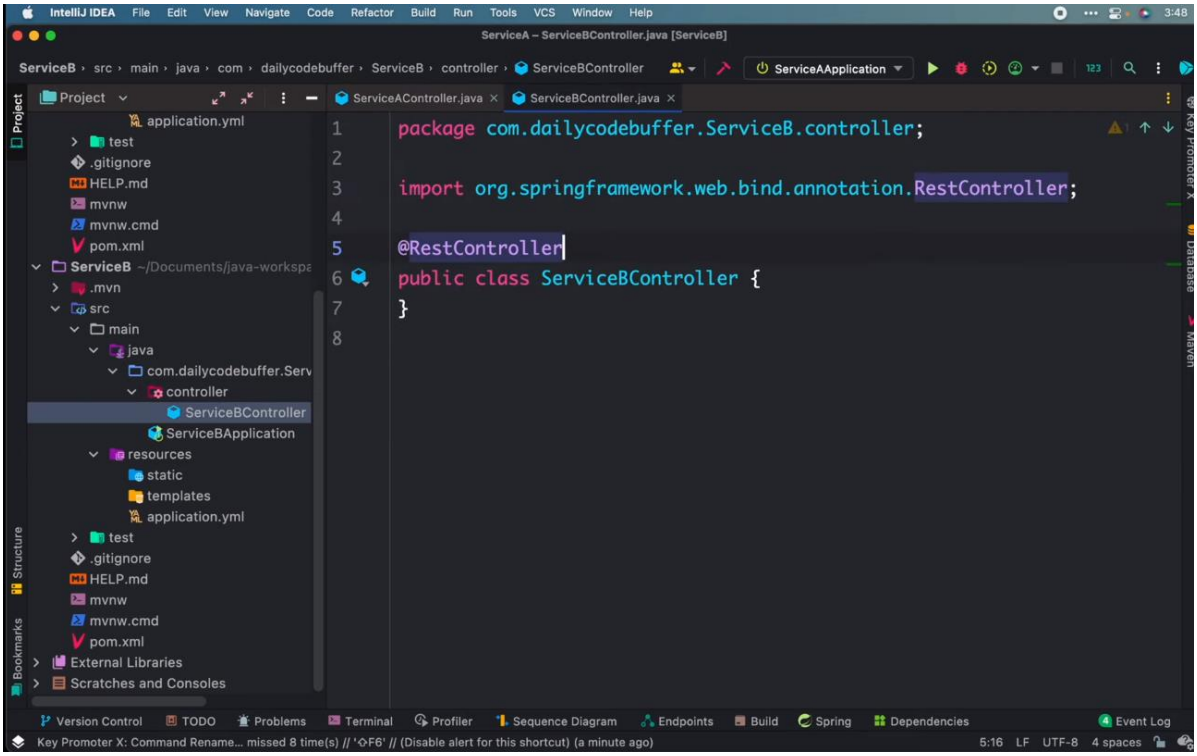

Resilience4j - Google Slides
Spring Initializr
Welcome to IntelliJ IDEA
Favorites
Recents
Applications
Documents
Desktop
Downloads
shabbir
Pictures
Music
ServiceA
angular-workspace
eclipse-workspace
flutter-workspace
GitHub
go-workspace
java-workspace
react-workspace
xcode-workspace
Zoom
code-samples-master
CQRS-demo
EMS
Event Driven
resilience4j
spring-security-tutorial
test
ServiceA
ServiceB
HELP.md
mvnw
mvnw.cmd
pom.xml
src

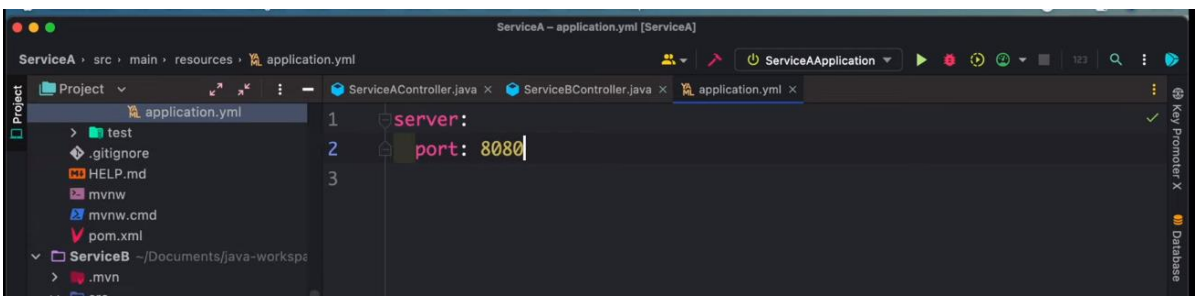

ServiceA
ServiceA › pom.xml
ServiceAApplication
Project
ServiceA ~/Documents/java-workspace
.mvn
src
.gitignore
HELP.md
mvnw
mvnw.cmd
pom.xml
External Libraries
Scratches and Consoles
Maven
ServiceA
Search Everywhere Double ⇧
Go to File ⇧⌘O

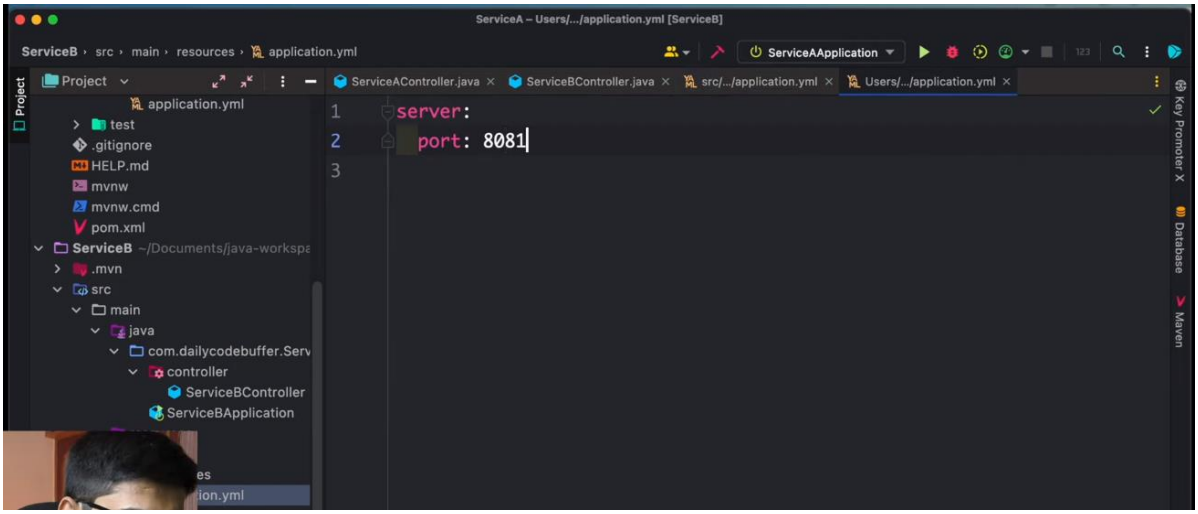

ServiceA
ServiceA › pom.xml
ServiceAApplication
Project
Maven
Favorites
Recents
Applications
Documents
Desktop
Downloads
shabbir
Pictures
Music
Movies
ServiceB
angular-workspace
eclipse-workspace
flutter-workspace
GitHub
go-workspace
java-workspace
react-workspace
xcode-workspace
Zoom
code-samples-master
CQRS-demo
EMS
Event Driven
resilience4j
spring-security-tutorial
test
ServiceA
ServiceB
HELP.md
mvnw
mvnw.cmd
pom.xml
src

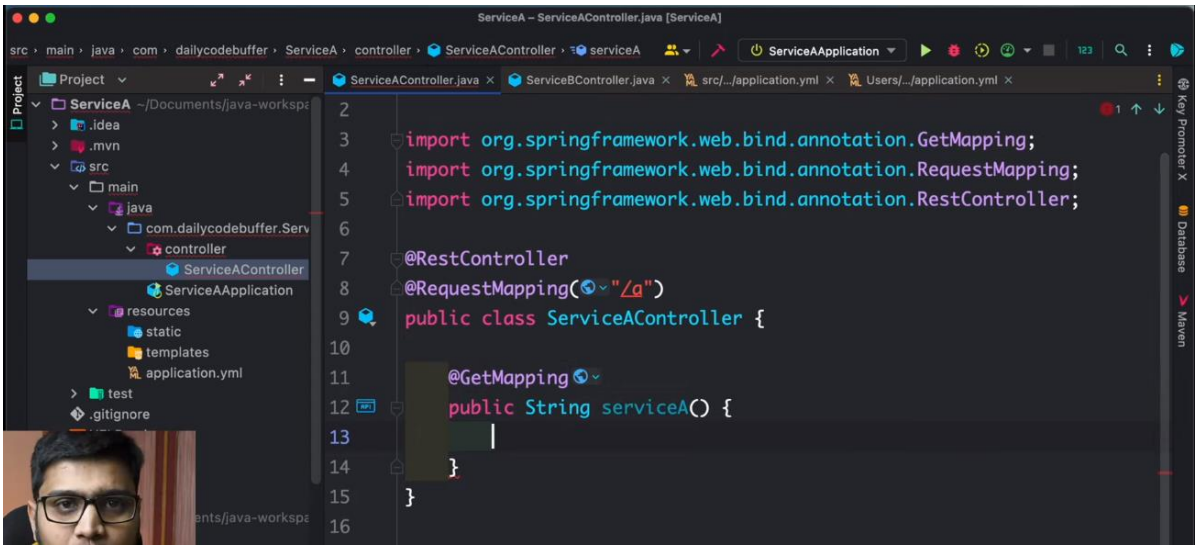

IntelliJ IDEA File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
3:47
ServiceA
ServiceB › src › main › resources
ServiceAApplication
Project
ServiceA ~/Documents/java-workspace
.mvn
src
main
java
resources
static
templates
application.yml
test
.gitignore
HELP.md
mvnw
mvnw.cmd
pom.xml
ServiceB ~/Documents/java-workspace
Search Everywhere Double ⇧
Go to File ⇧⌘O
Recent Files ⌘E
Navigation Bar ⌘↑
Drop files here to open them
Version Control
TODO
Problems
Terminal
Profiler
Sequence Diagram
Endpoints
Build
Spring
Dependencies
Event Log
Key Promoter X: Command Rename... missed 8 time(s) // '⇧F6' // (Disable alert for this shortcut) (moments ago)


ServiceA – ServiceAController.java [ServiceA]
ServiceA › src › main › java › com › dailycodebuffer › ServiceA › controller › ServiceAController
ServiceAApplication
ServiceAController.java
package com.dailycodebuffer.ServiceA.controller;
import org.springframework.web.bind.annotation.RestController;
@RestController
public class ServiceAController {
}
Project
ServiceAApplication
resources
static
templates
application.yml
test
.gitignore
HELP.md
mvnw
mvnw.cmd
pom.xml
ServiceB ~/Documents/java-workspa
.mvn
src
main
java
resources
static
templates
application.yml
test
.gitignore
HELP.md
mvnw
mvnw.cmd
pom.xml
External Libraries
Scratches and Consoles
Version Control
TODO
Problems
Terminal
Profiler
Sequence Diagram
Endpoints
Build
Spring
Dependencies
Event Log

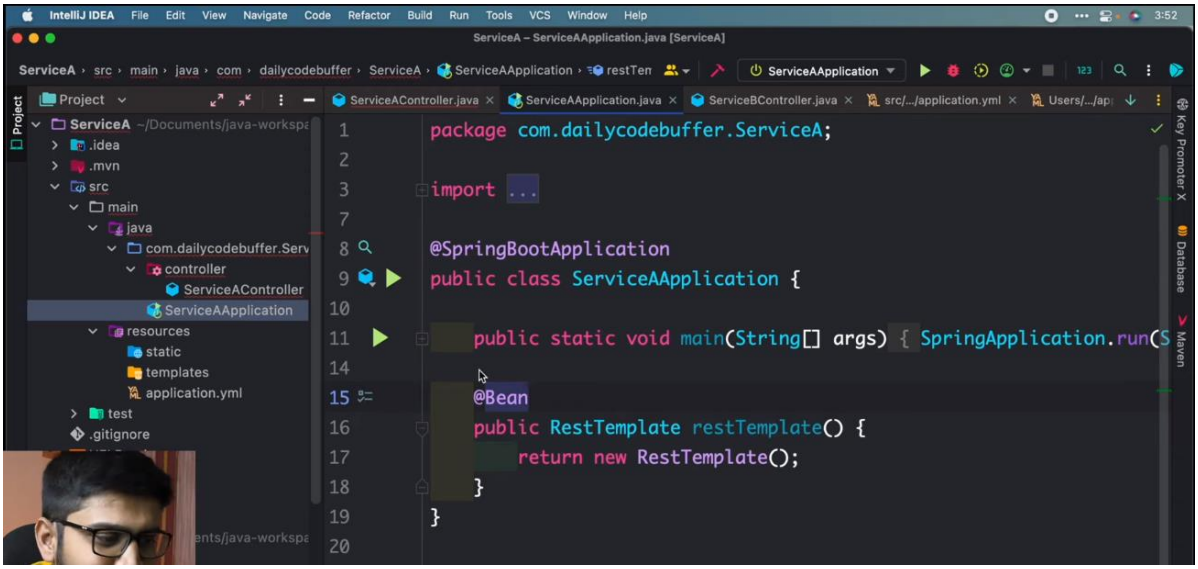

ServiceB – ServiceBController.java [ServiceB]
ServiceB › src › main › java › com › dailycodebuffer › ServiceB › controller › ServiceBController
ServiceAApplication
ServiceAController.java
ServiceBController.java
package com.dailycodebuffer.ServiceB.controller;
import org.springframework.web.bind.annotation.RestController;
@RestController
public class ServiceBController {
}
Project
application.yml
test
.gitignore
HELP.md
mvnw
mvnw.cmd
pom.xml
ServiceB ~/Documents/java-workspa
.mvn
src
main
java
com.dailycodebuffer.Serv
controller
ServiceBController
ServiceBApplication
resources
static
templates
application.yml
test
.gitignore
HELP.md
mvnw
mvnw.cmd
pom.xml
External Libraries
Scratches and Consoles
Version Control
TODO
Problems
Terminal
Profiler
Sequence Diagram
Endpoints
Build
Spring
Dependencies
Event Log
Key Promoter X: Command Rename... missed 8 time(s) // '⌥F6' // (Disable alert for this shortcut) (a minute ago)
5:16
LF
UTF-8
4 spaces


ServiceA – application.yml [ServiceA]
ServiceA › src › main › resources › application.yml
ServiceAApplication
ServiceAController.java
ServiceBController.java
application.yml
server:
port: 8080
Project
application.yml
test
.gitignore
HELP.md
mvnw
mvnw.cmd
pom.xml
ServiceB ~/Documents/java-workspa
.mvn

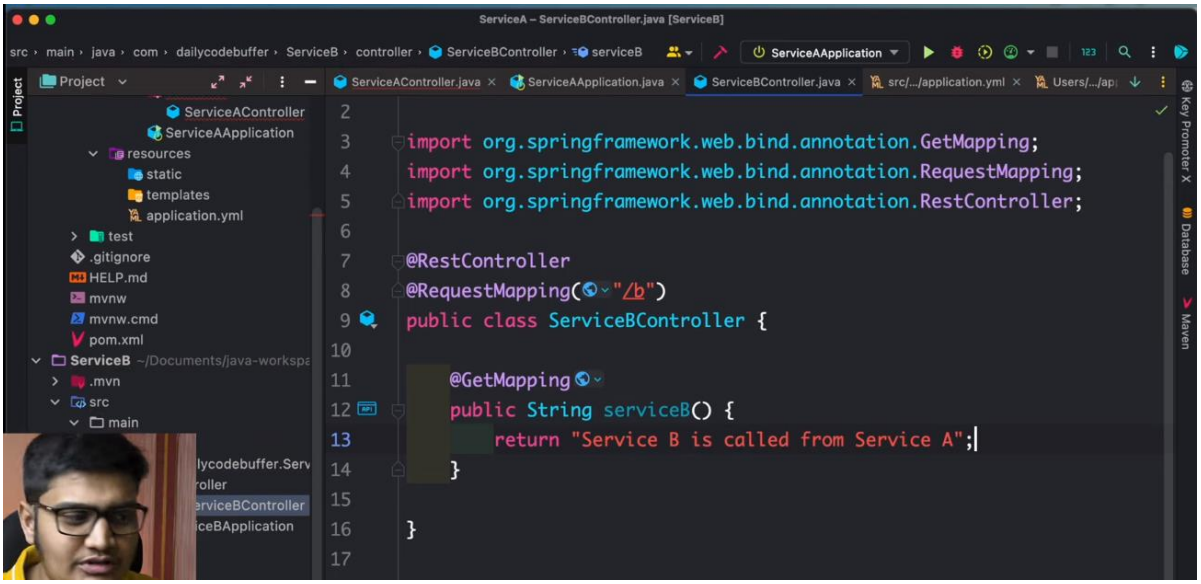


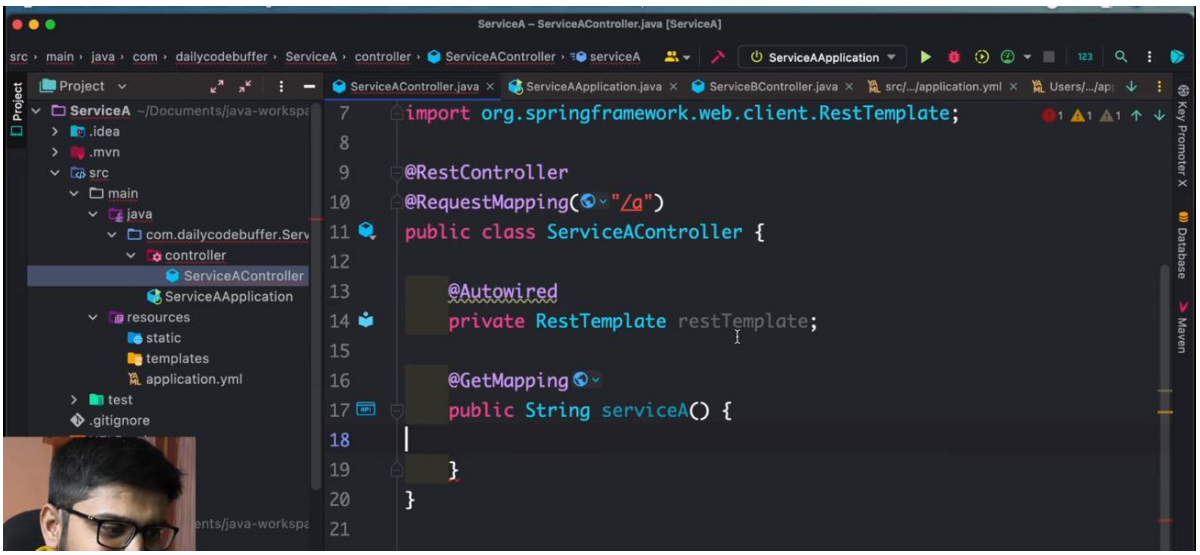


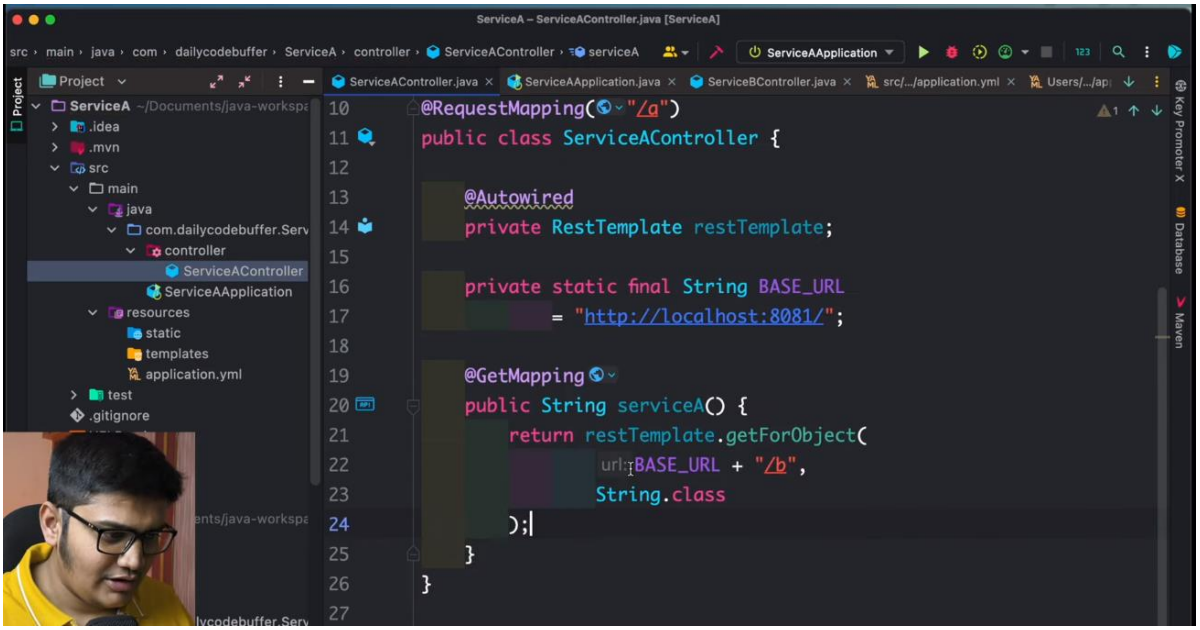
Let us now implement the circuit breaker, retry and rate limiter in the services, we want to call service B from service A using the RestTemplate or WebClient by creating the bean and autowiring it.

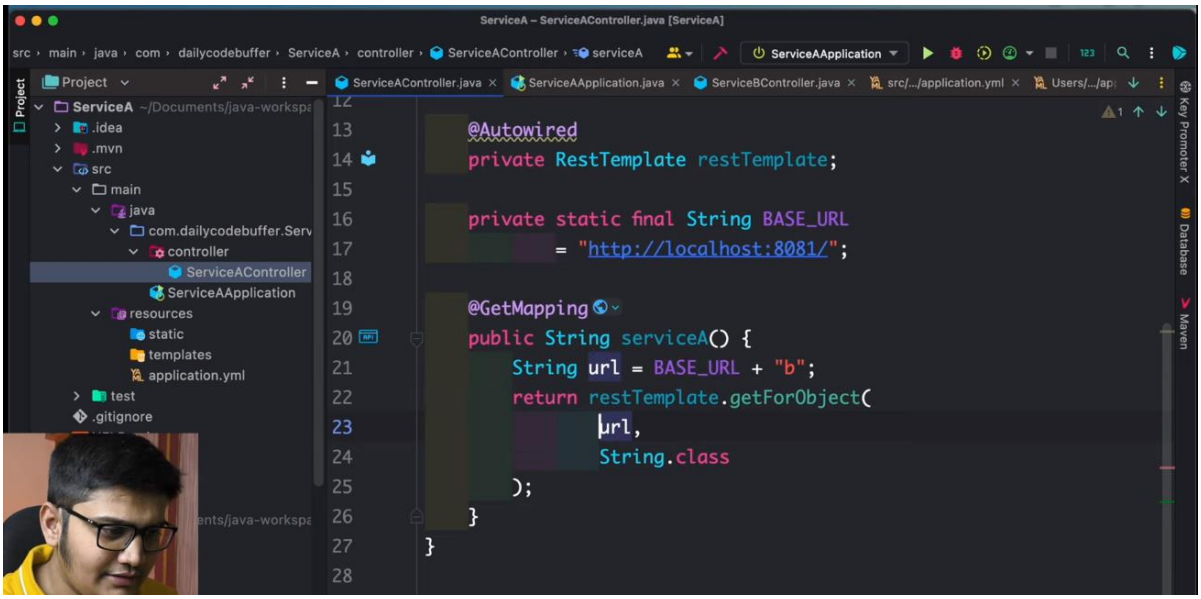


We can now go use this bean in our controller below.

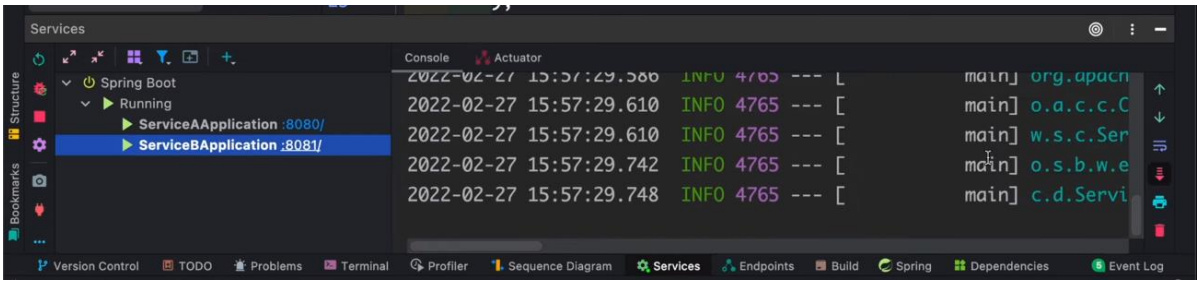

ServiceA – ServiceBController.java [ServiceB]
src › main › java › com › dailycodebuffer › ServiceB › controller › ServiceBController › serviceB
ServiceAApplication
Project
ServiceAController
ServiceAApplication
resources
static
templates
application.yml
test
.gitignore
HELP.md
mvnw
mvnw.cmd
pom.xml
ServiceB ~/Documents/java-workspa
.mvn
src
main
ServiceAController.java
ServiceAApplication.java
ServiceBController.java
src/.../application.yml
Users/.../ap
2
3 import org.springframework.web.bind.annotation.GetMapping;
4 import org.springframework.web.bind.annotation.RequestMapping;
5 import org.springframework.web.bind.annotation.RestController;
6
7 @RestController
8 @RequestMapping("/b")
9 public class ServiceBController {
10
11 @GetMapping
12 public String serviceB() {
13 return "Service B is called from Service A";
14 }
15
16 }
17
Key Promoter X
Database
Maven

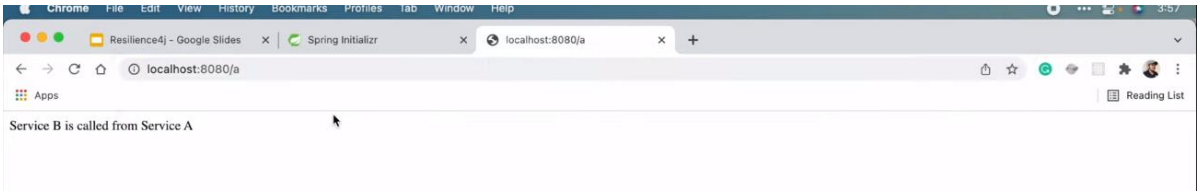

ServiceA – ServiceAController.java [ServiceA]
src › main › java › com › dailycodebuffer › ServiceA › controller › ServiceAController › serviceA
ServiceAApplication
Project
ServiceA ~/Documents/java-workspa
.idea
.mvn
src
main
java
com.dailycodebuffer.Serv
controller
ServiceAController
ServiceAApplication
resources
static
templates
application.yml
test
.gitignore
ServiceAController.java
ServiceAApplication.java
ServiceBController.java
src/.../application.yml
Users/.../ap
7 import org.springframework.web.client.RestTemplate;
8
9 @RestController
10 @RequestMapping("/a")
11 public class ServiceAController {
12
13 @Autowired
14 private RestTemplate restTemplate;
15
16 @GetMapping
17 public String serviceA() {
18
19 }
20 }
21
Key Promoter X
Database
Maven

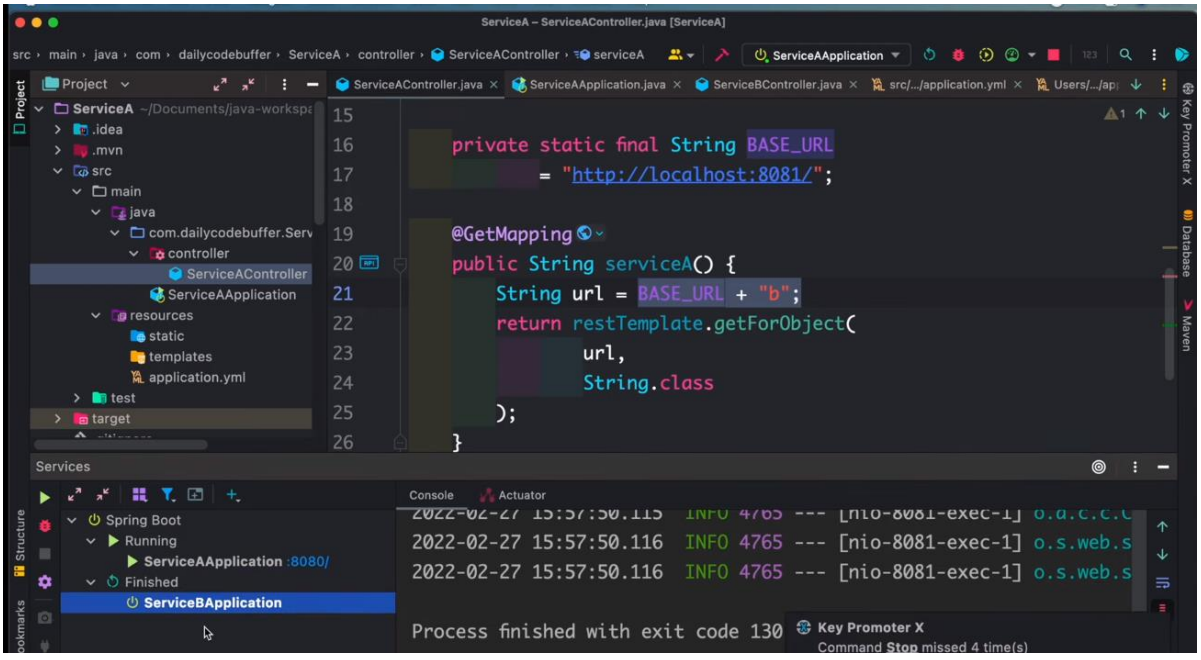

ServiceA – ServiceAController.java [ServiceA]
src › main › java › com › dailycodebuffer › ServiceA › controller › ServiceAController › serviceA
ServiceAApplication
Project
ServiceA ~/Documents/java-workspa
.idea
.mvn
src
main
java
com.dailycodebuffer.Serv
controller
ServiceAController
ServiceAApplication
resources
static
templates
application.yml
test
.gitignore
ServiceAController.java
ServiceAApplication.java
ServiceBController.java
src/.../application.yml
Users/.../ap
10 @RequestMapping("/a")
11 public class ServiceAController {
12
13 @Autowired
14 private RestTemplate restTemplate;
15
16 private static final String BASE_URL
17 = "http://localhost:8081/";
18
19 @GetMapping
20 public String serviceA() {
21 return restTemplate.getForObject(
22 url: BASE_URL + "/b",
23 String.class
24);
25 }
26 }
27
Key Promoter X
Database
Maven



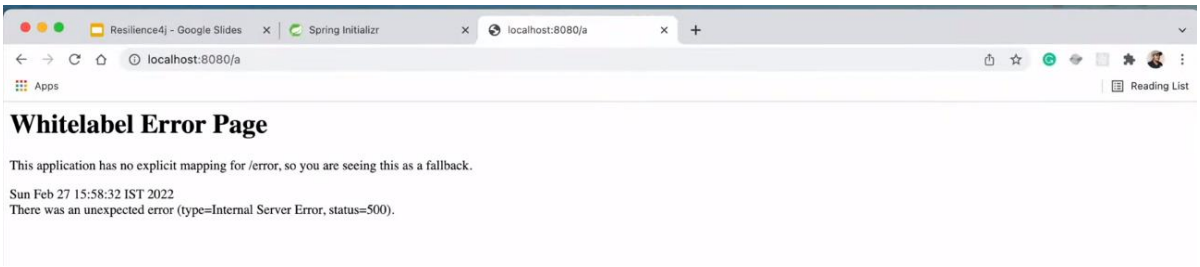
We are now calling service B from service A. then we run the services



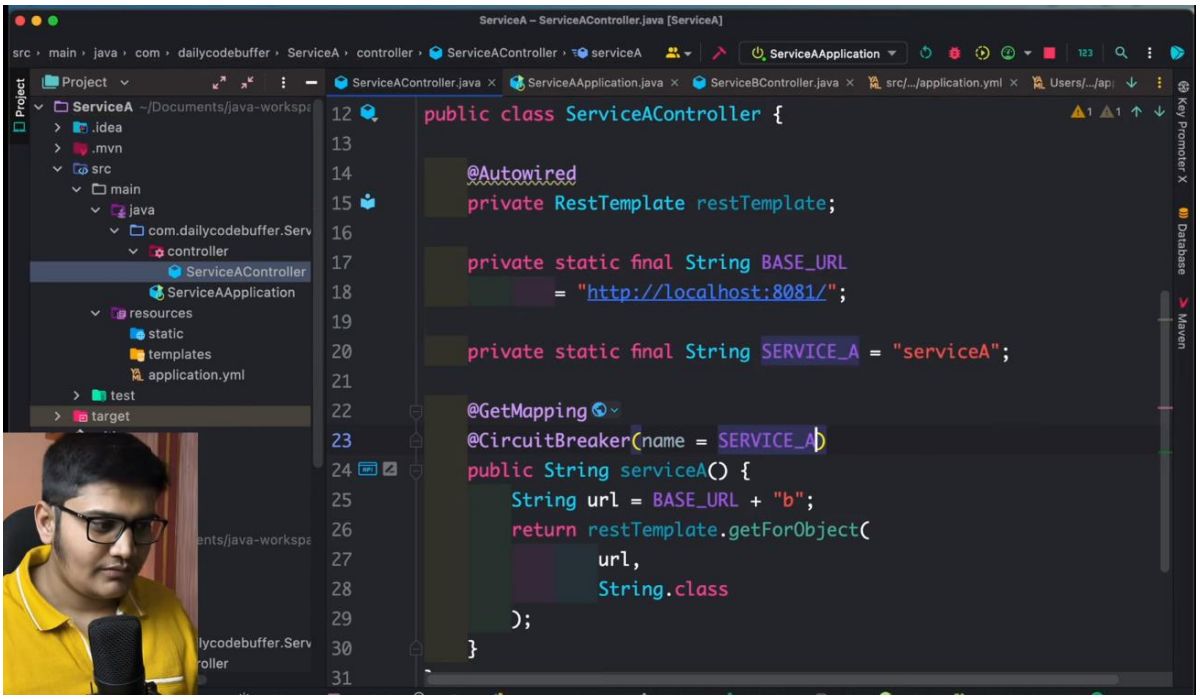




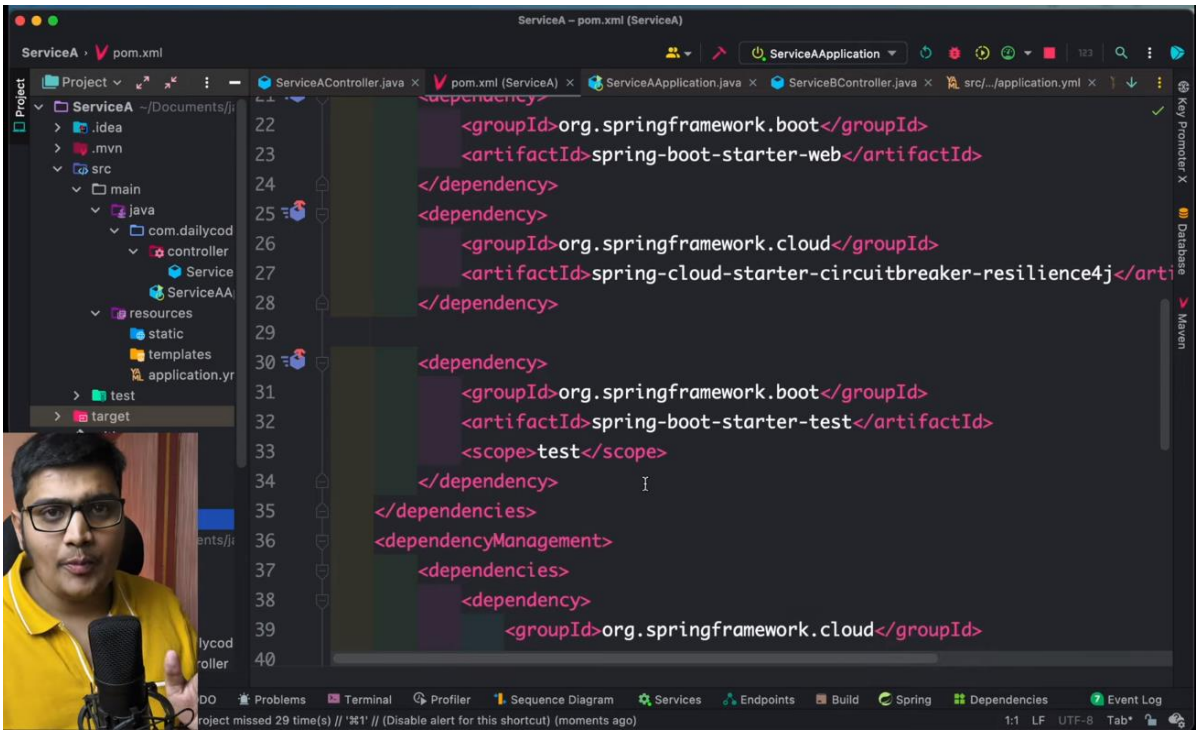
Next, let us kill service B above, we will now get the response below



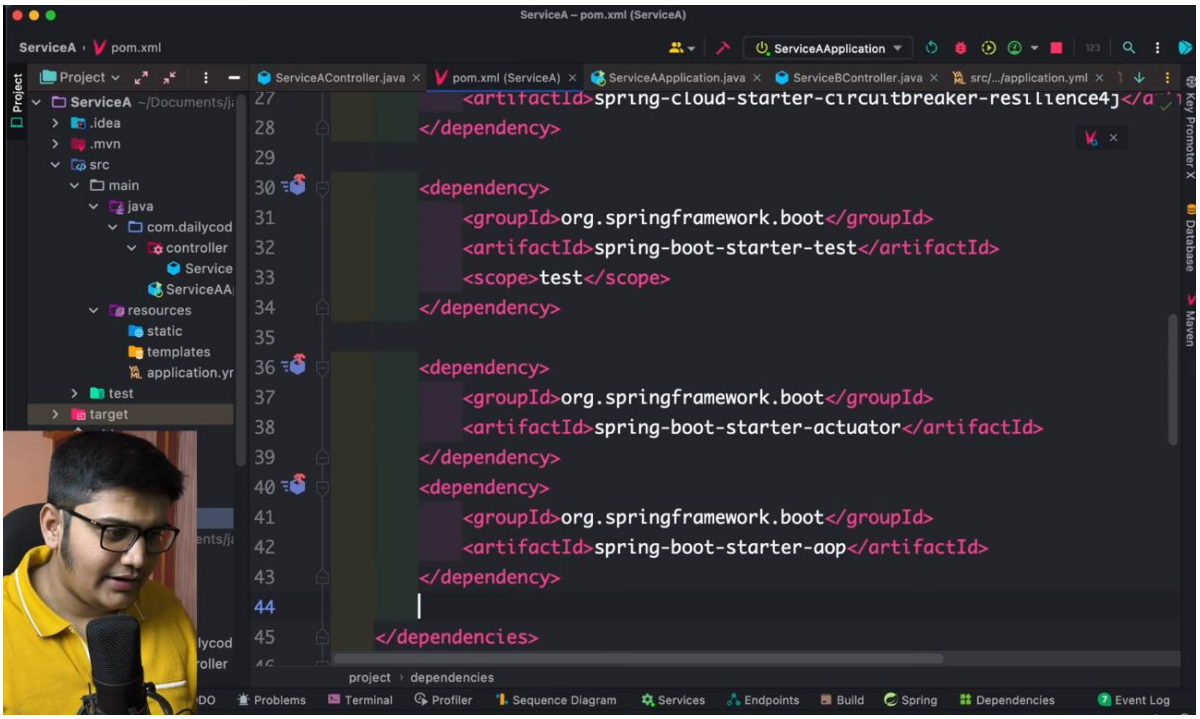
Let us now implement the circuit breaker pattern for the scenario where service B is down to use a fallback method

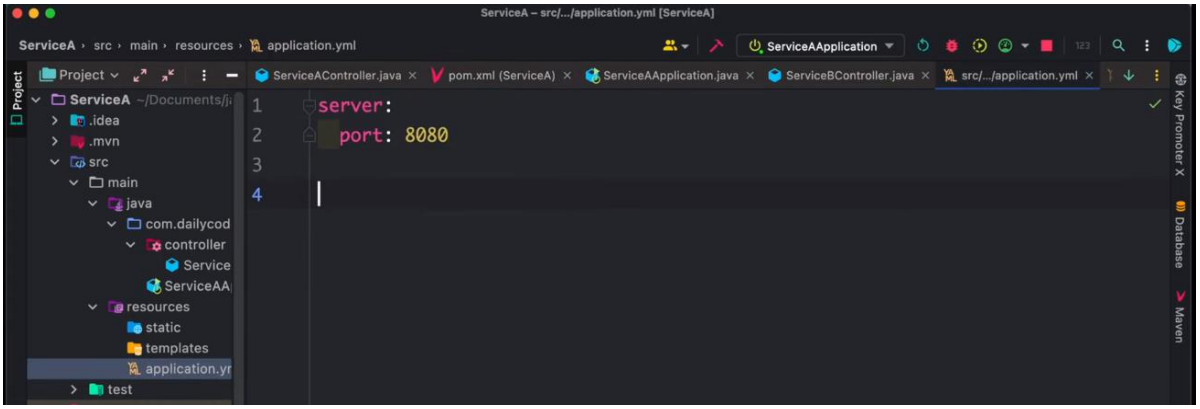


We add the **service_name** and the **@CircuitBreaker** annotation from Resilience4j to our service A endpoints as above.

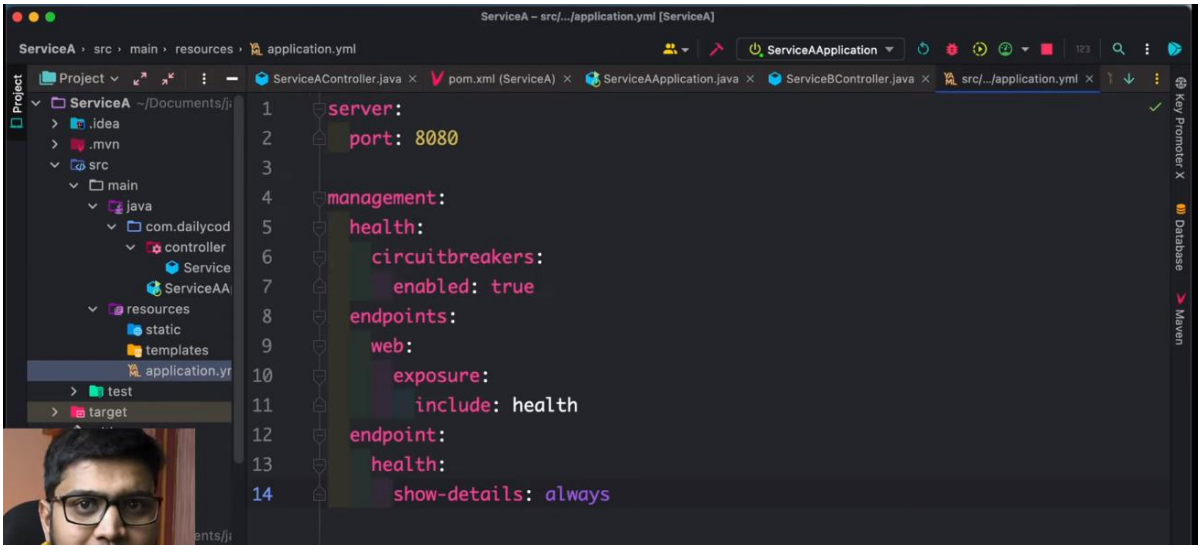


To service A, we need to add the AOP dependency and the actuator dependency as below

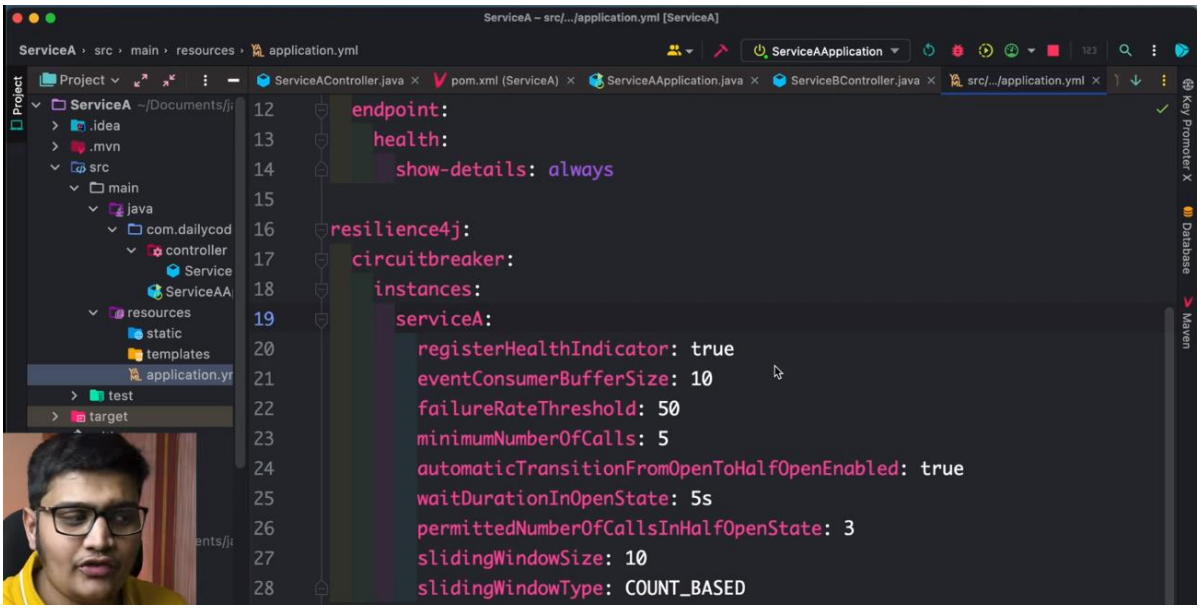




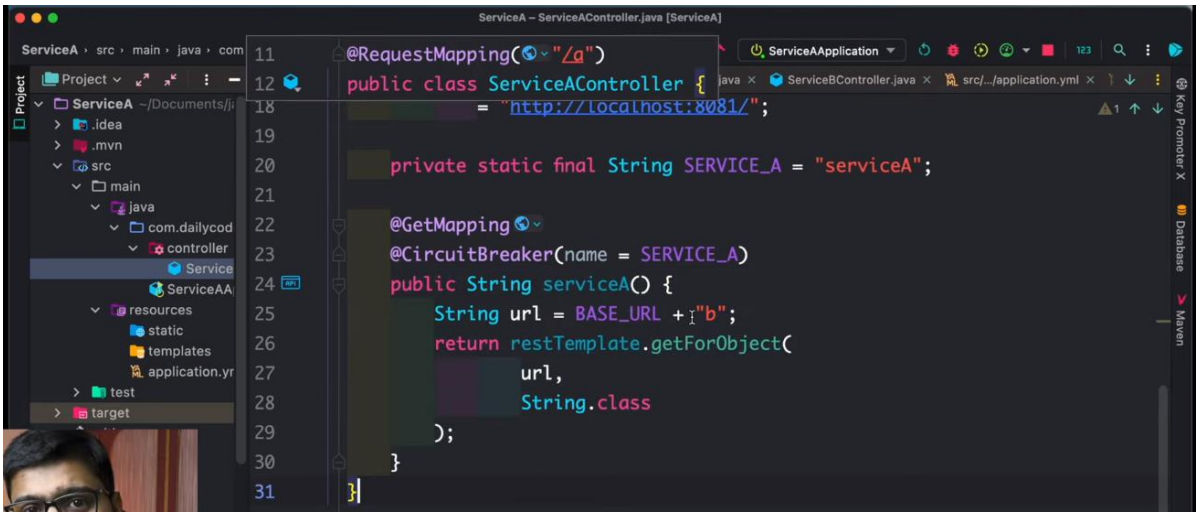
Next, we then add the circuit breaker and fallback configurations to our YAML file as below.

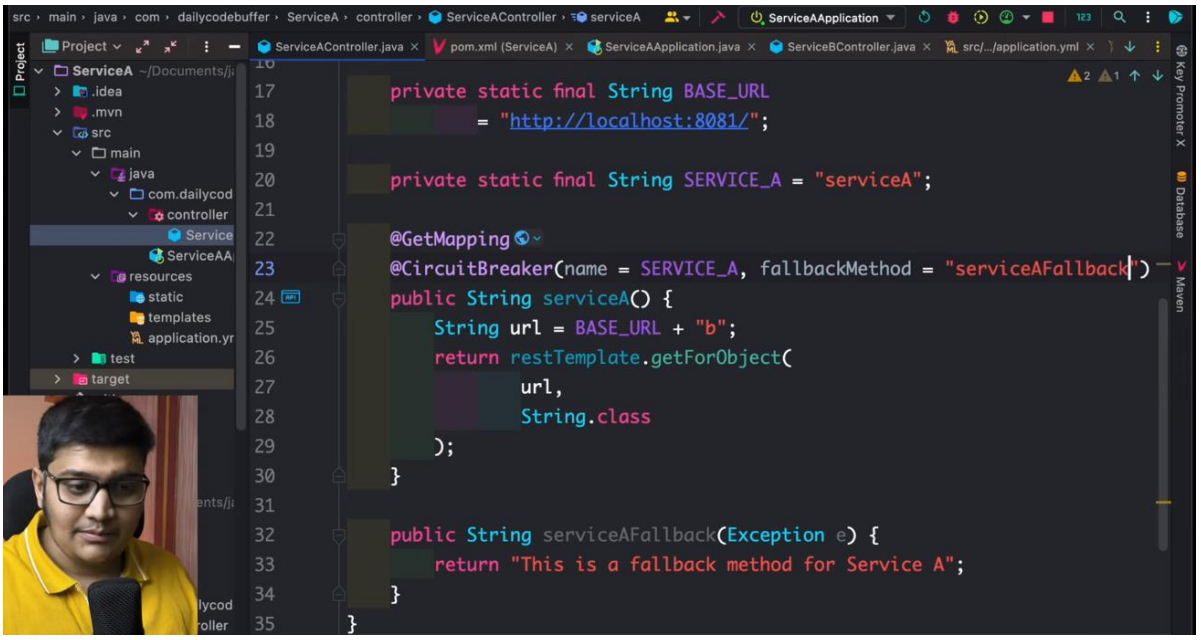


We enable the management endpoints for our actuators as above

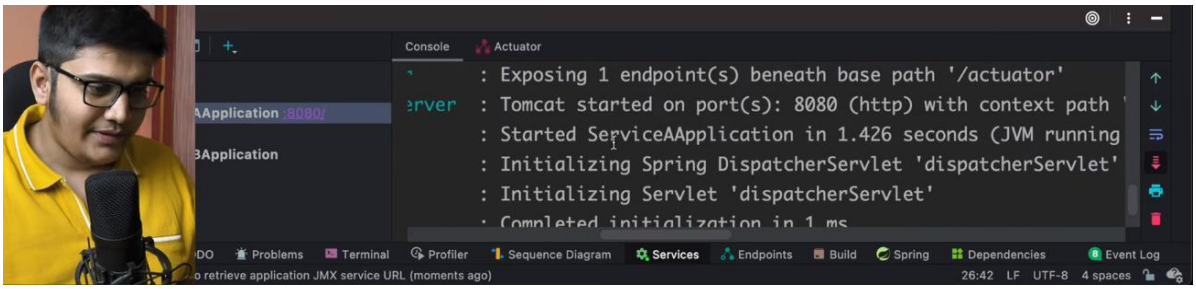


We then add the configuration for our Resilience4j circuit breaker for all our instances above

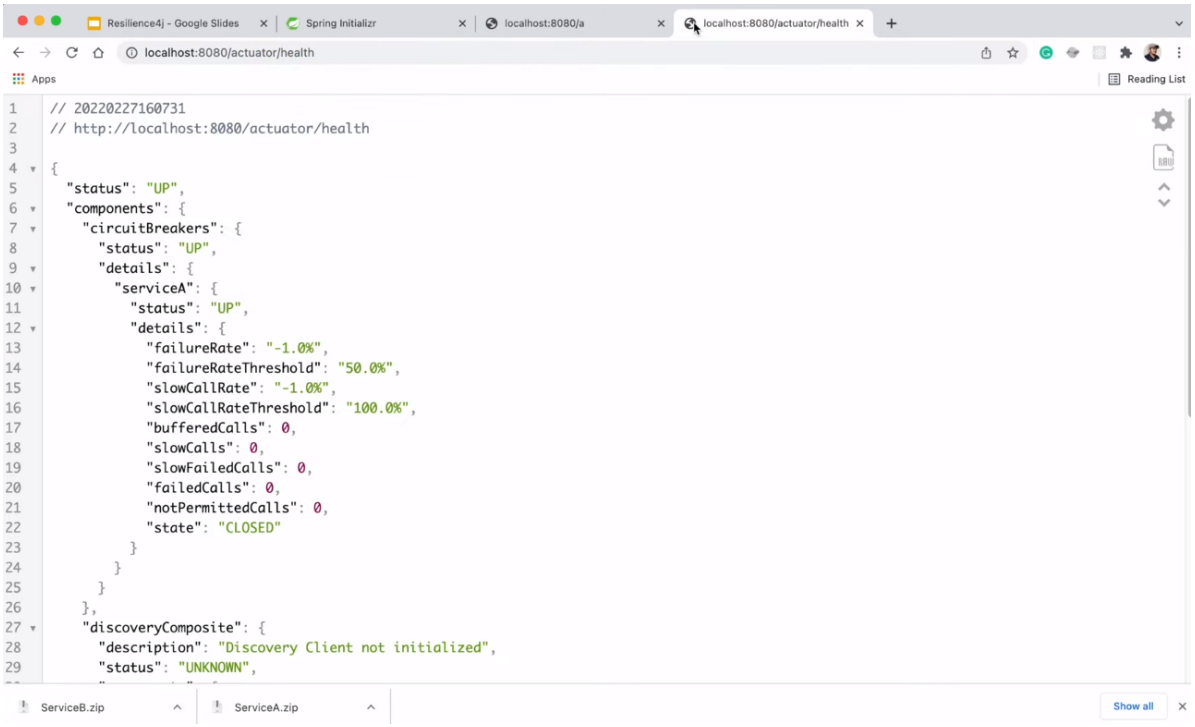


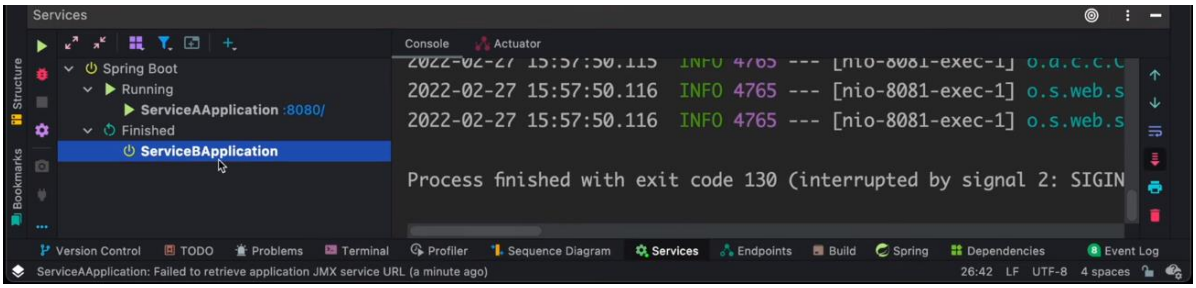


We create our fallback method that injects the exception and use it in the circuitbreaker method as above

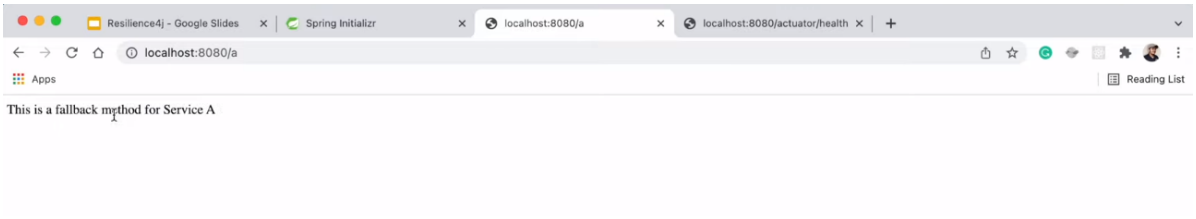


Restart app

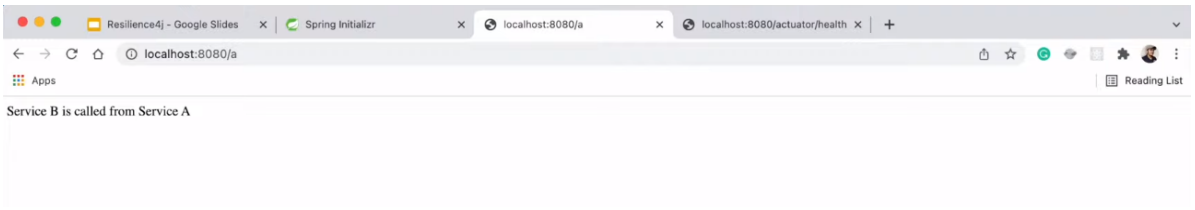




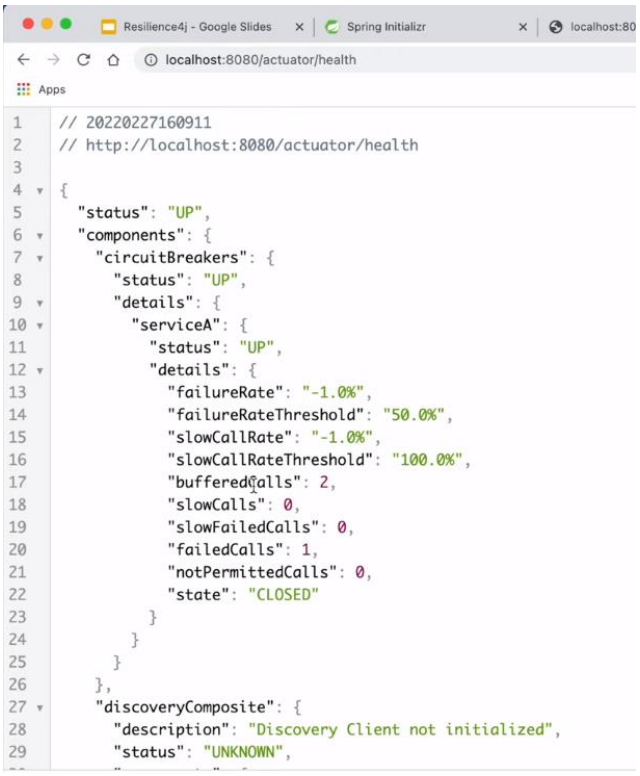
Note that service B is currently down

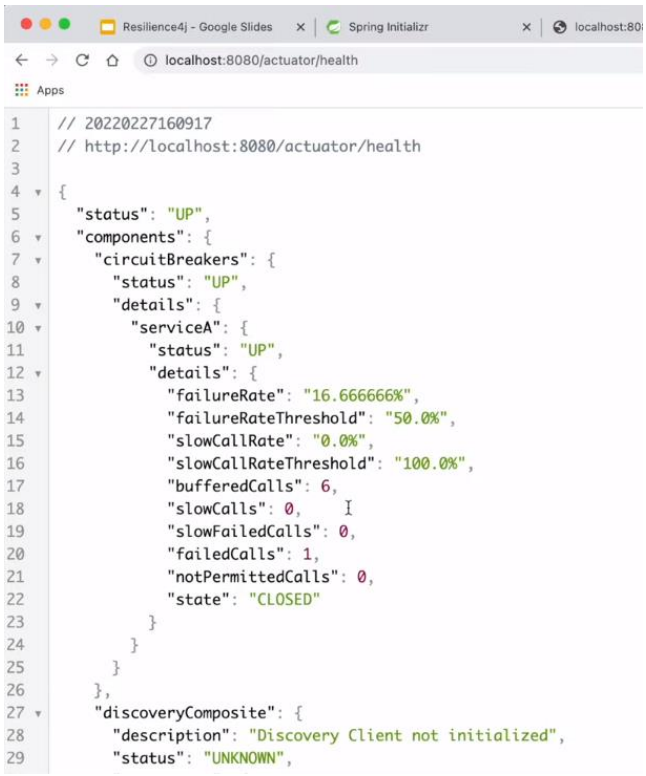


We can now see the fallback method response instead of a 404 page

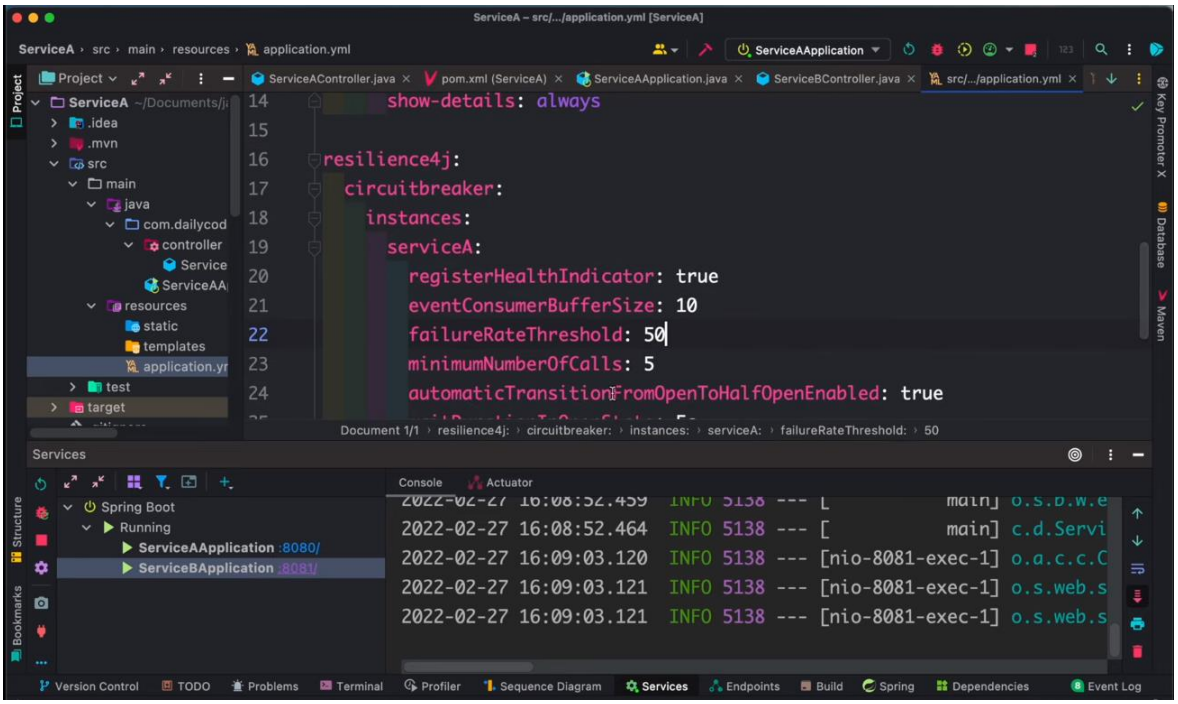


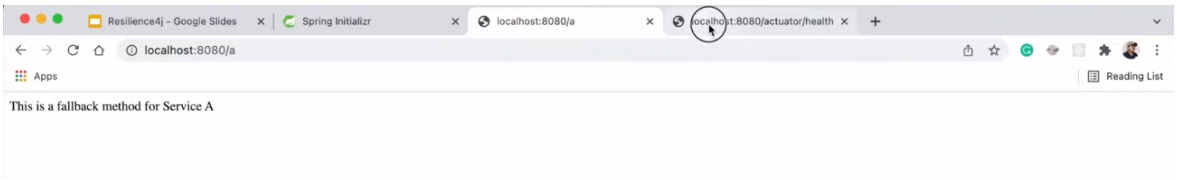
Restart service B again

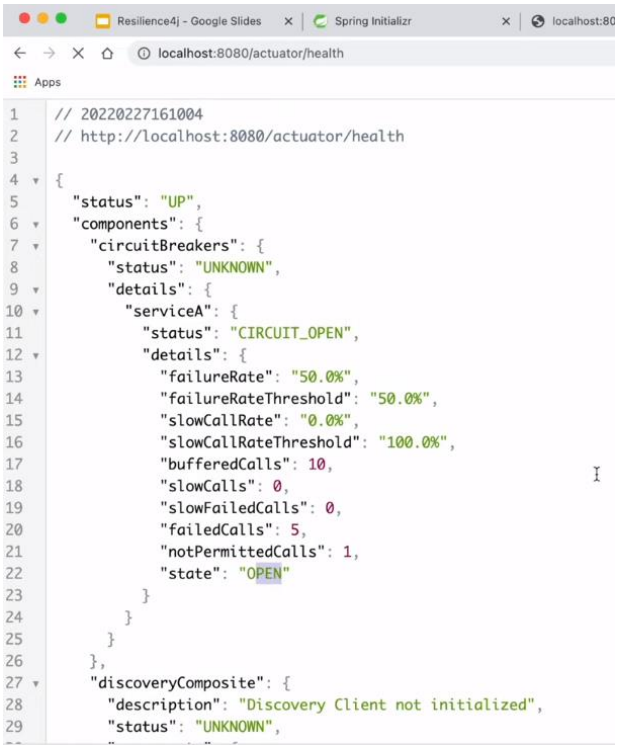


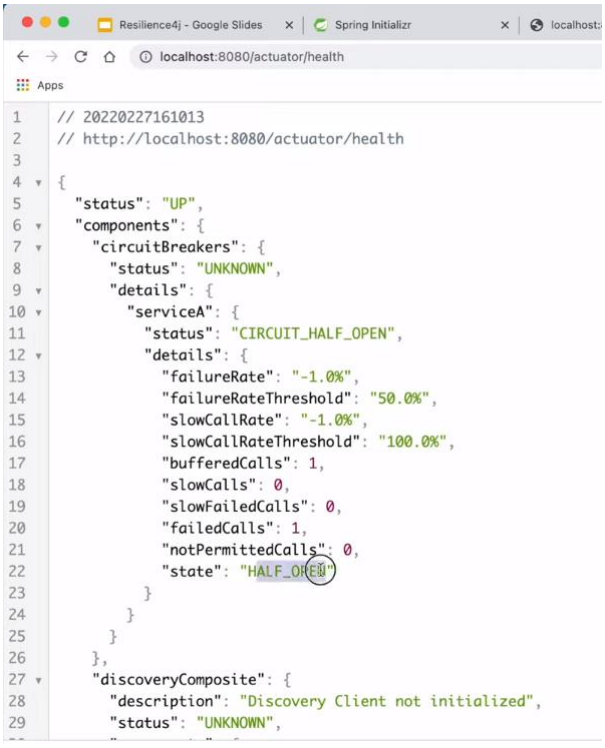


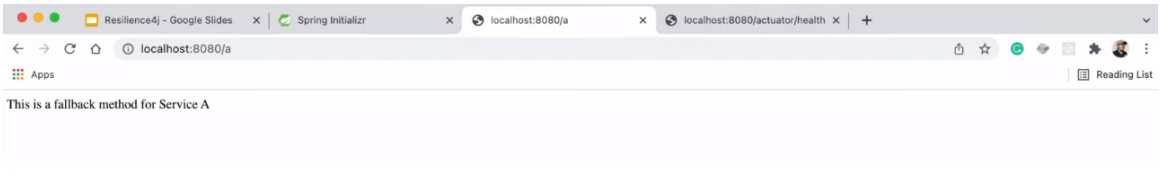
Note that

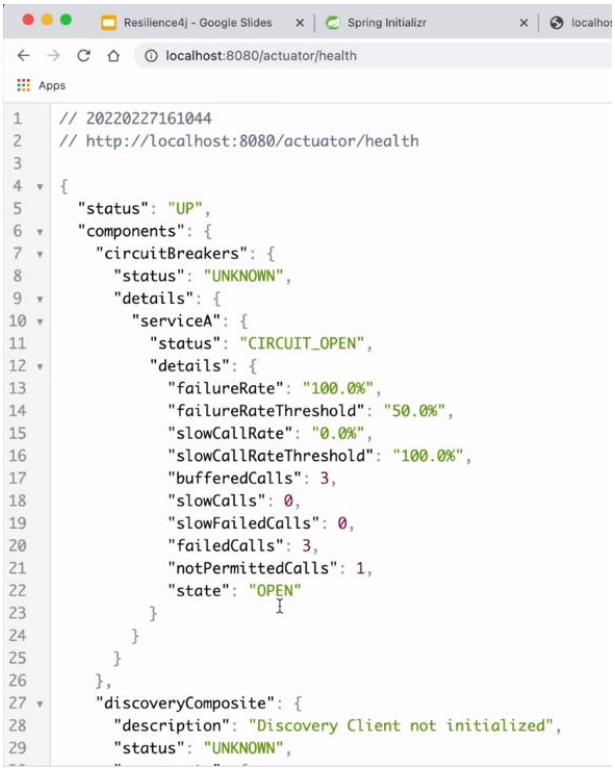


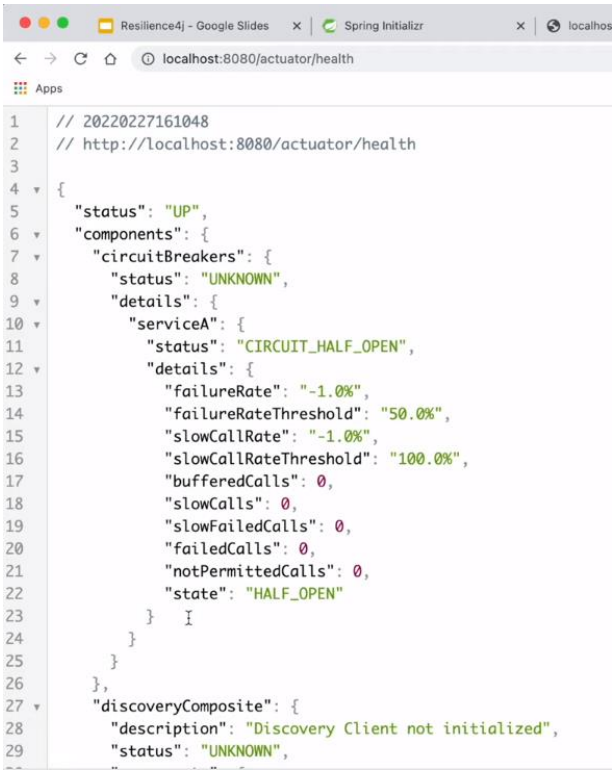

Resilience4j - Google Slides
Spring Initializr
localhost:8080/a
localhost:8080/actuator/health
localhost:8080/a
Apps
Reading List
This is a fallback method for Service A

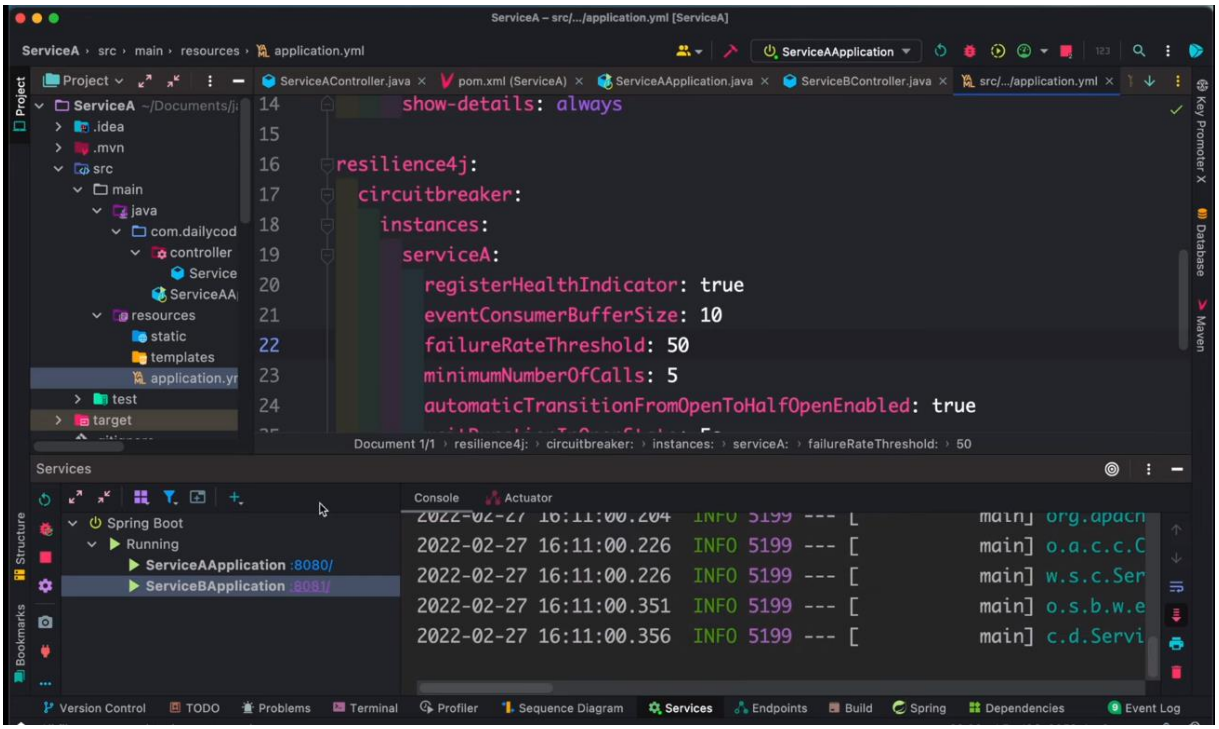

Resilience4j - Google Slides
Spring Initializr
localhost:8080/actuator/health
Apps
// 20220227161004
// http://localhost:8080/actuator/health
{
"status": "UP",
"components": {
"circuitBreakers": {
"status": "UNKNOWN",
"details": {
"serviceA": {
"status": "CIRCUIT_OPEN",
"details": {
"failureRate": "50.0%",
"failureRateThreshold": "50.0%",
"slowCallRate": "0.0%",
"slowCallRateThreshold": "100.0%",
"bufferedCalls": 10,
"slowCalls": 0,
"slowFailedCalls": 0,
"failedCalls": 5,
"notPermittedCalls": 1,
"state": "OPEN"
}
}
}
},
"discoveryComposite": {
"description": "Discovery Client not initialized",
"status": "UNKNOWN",

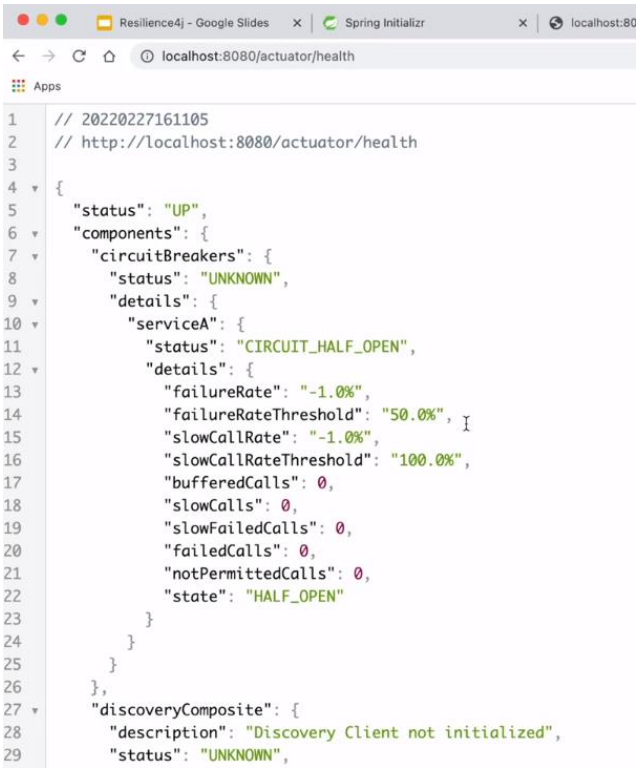

Resilience4j - Google Slides
Spring Initializr
localhost:8080/actuator/health
Apps
// 20220227161013
// http://localhost:8080/actuator/health
{
"status": "UP",
"components": {
"circuitBreakers": {
"status": "UNKNOWN",
"details": {
"serviceA": {
"status": "CIRCUIT_HALF_OPEN",
"details": {
"failureRate": "-1.0%",
"failureRateThreshold": "50.0%",
"slowCallRate": "-1.0%",
"slowCallRateThreshold": "100.0%",
"bufferedCalls": 1,
"slowCalls": 0,
"slowFailedCalls": 0,
"failedCalls": 1,
"notPermittedCalls": 0,
"state": "HALF_OPEN"
}
}
}
},
"discoveryComposite": {
"description": "Discovery Client not initialized",
"status": "UNKNOWN",

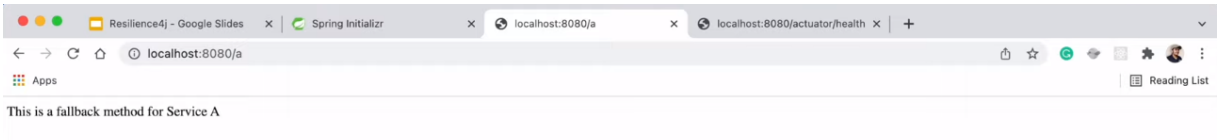

Resilience4j - Google Slides
Spring Initializr
localhost:8080/a
localhost:8080/actuator/health
localhost:8080/a
Apps
Reading List
This is a fallback method for Service A

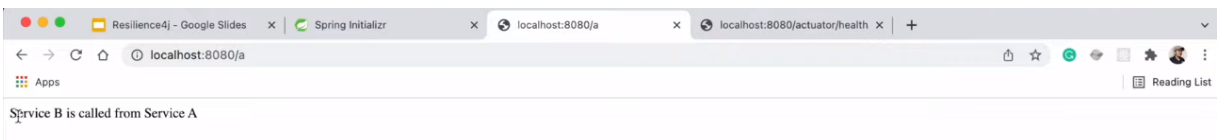

Resilience4j - Google Slides
Spring Initializr
localhost:8080/actuator/health
Apps
// 20220227161044
// http://localhost:8080/actuator/health
{
"status": "UP",
"components": {
"circuitBreakers": {
"status": "UNKNOWN",
"details": {
"serviceA": {
"status": "CIRCUIT_OPEN",
"details": {
"failureRate": "100.0%",
"failureRateThreshold": "50.0%",
"slowCallRate": "0.0%",
"slowCallRateThreshold": "100.0%",
"bufferedCalls": 3,
"slowCalls": 0,
"slowFailedCalls": 0,
"failedCalls": 3,
"notPermittedCalls": 1,
"state": "OPEN"
}
}
}
},
"discoveryComposite": {
"description": "Discovery Client not initialized",
"status": "UNKNOWN",

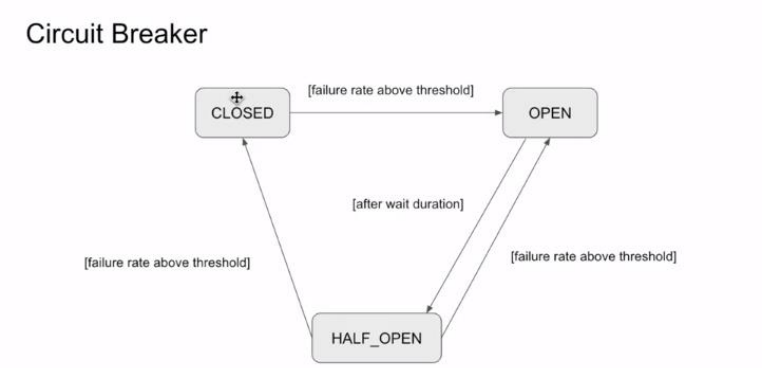

Resilience4j - Google Slides
Spring Initializr
localhost:8080/actuator/health
Apps
// 20220227161048
// http://localhost:8080/actuator/health
{
"status": "UP",
"components": {
"circuitBreakers": {
"status": "UNKNOWN",
"details": {
"serviceA": {
"status": "CIRCUIT_HALF_OPEN",
"details": {
"failureRate": "-1.0%",
"failureRateThreshold": "50.0%",
"slowCallRate": "-1.0%",
"slowCallRateThreshold": "100.0%",
"bufferedCalls": 0,
"slowCalls": 0,
"slowFailedCalls": 0,
"failedCalls": 0,
"notPermittedCalls": 0,
"state": "HALF_OPEN"
}
}
}
},
"discoveryComposite": {
"description": "Discovery Client not initialized",
"status": "UNKNOWN",


ServiceA – src/.../application.yml [ServiceA]
ServiceA › src › main › resources › application.yml
ServiceAApplication
ServiceAController.java
pom.xml (ServiceA)
ServiceAApplication.java
ServiceBController.java
src/.../application.yml
14 show-details: always
15
16 resilience4j:
17 circuitbreaker:
18 instances:
19 serviceA:
20 registerHealthIndicator: true
21 eventConsumerBufferSize: 10
22 failureRateThreshold: 50
23 minimumNumberOfCalls: 5
24 automaticTransitionFromOpenToHalfOpenEnabled: true
Document 1/1 › resilience4j: › circuitbreaker: › instances: › serviceA: › failureRateThreshold: › 50
Services
Spring Boot
Running
ServiceAApplication :8080/
ServiceBApplication :8081/
Console
Actuator
2022-02-27 16:11:00.226 INFO 5199 --- [main] o.a.c.c.C
2022-02-27 16:11:00.226 INFO 5199 --- [main] w.s.c.Ser
2022-02-27 16:11:00.351 INFO 5199 --- [main] o.s.b.w.e
2022-02-27 16:11:00.356 INFO 5199 --- [main] c.d.Servi
Version Control
TODO
Problems
Terminal
Profiler
Sequence Diagram
Services
Endpoints
Build
Spring
Dependencies
Event Log

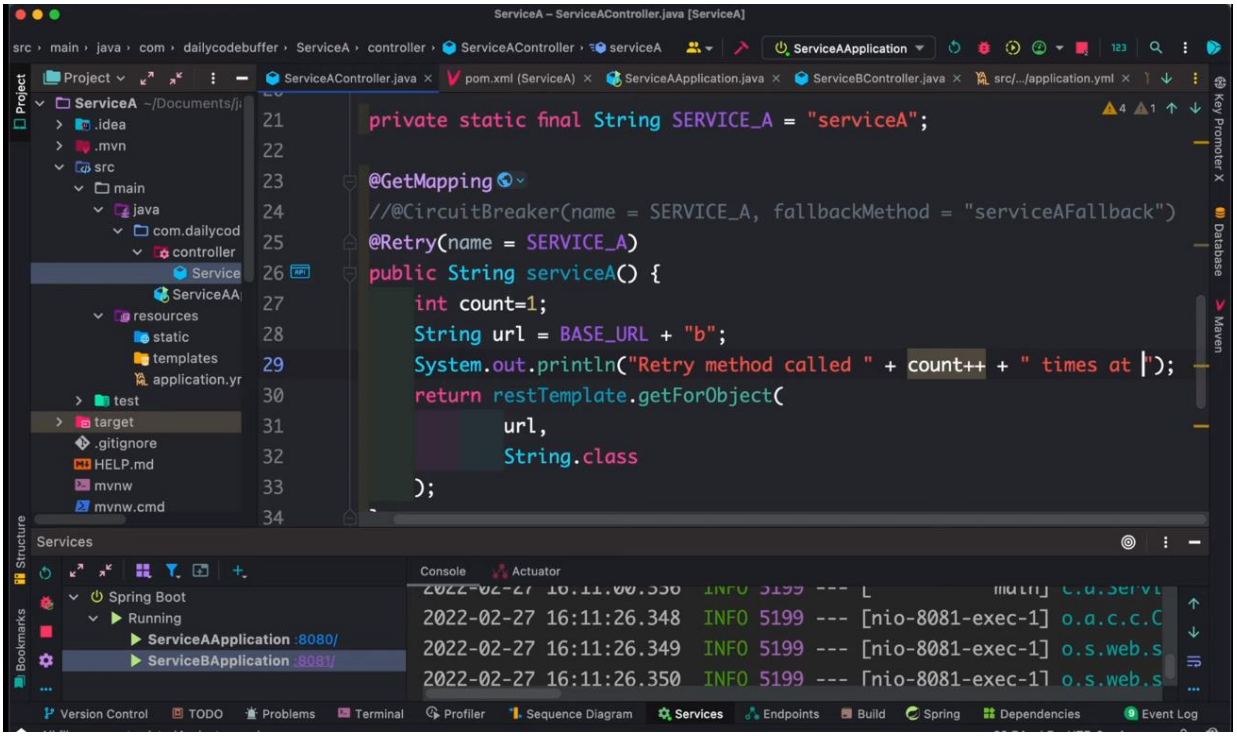

Resilience4j - Google Slides
Spring Initializr
localhost:8080/actuator/health
Apps
// 20220227161105
// http://localhost:8080/actuator/health
{
"status": "UP",
"components": {
"circuitBreakers": {
"status": "UNKNOWN",
"details": {
"serviceA": {
"status": "CIRCUIT_HALF_OPEN",
"details": {
"failureRate": "-1.0%",
"failureRateThreshold": "50.0%",
"slowCallRate": "-1.0%",
"slowCallRateThreshold": "100.0%",
"bufferedCalls": 0,
"slowCalls": 0,
"slowFailedCalls": 0,
"failedCalls": 0,
"notPermittedCalls": 0,
"state": "HALF_OPEN"
}
}
}
},
"discoveryComposite": {
"description": "Discovery Client not initialized",
"status": "UNKNOWN",


Resilience4j - Google Slides
Spring Initializr
localhost:8080/a
localhost:8080/actuator/health
Apps
Reading List
This is a fallback method for Service A

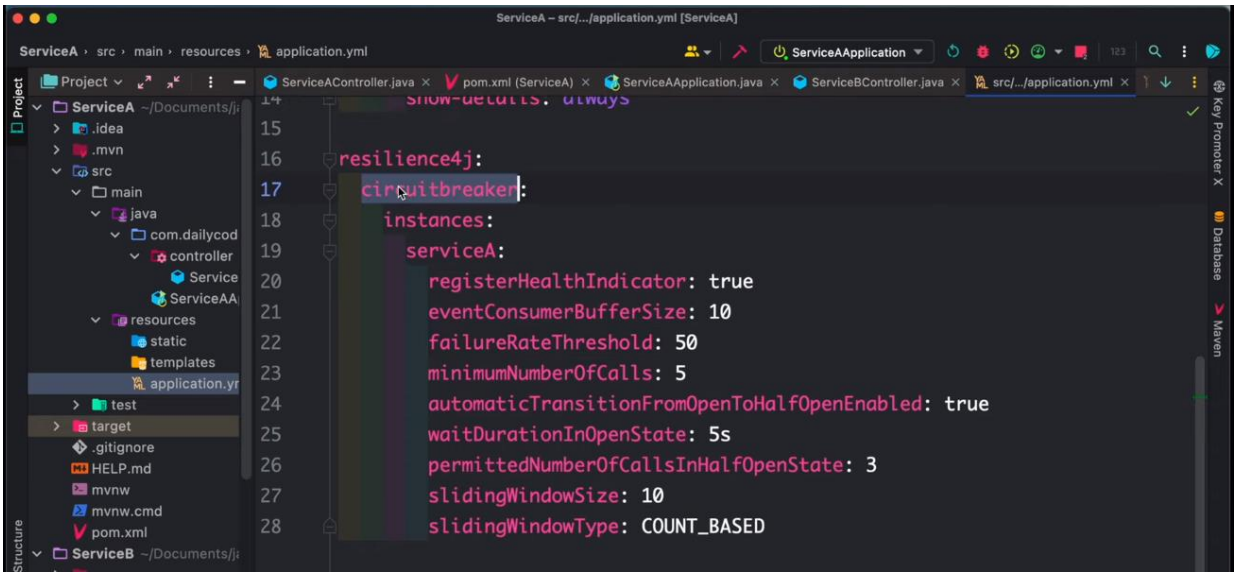

Resilience4j - Google Slides
Spring Initializr
localhost:8080/a
localhost:8080/actuator/health
Apps
Reading List
Service B is called from Service A

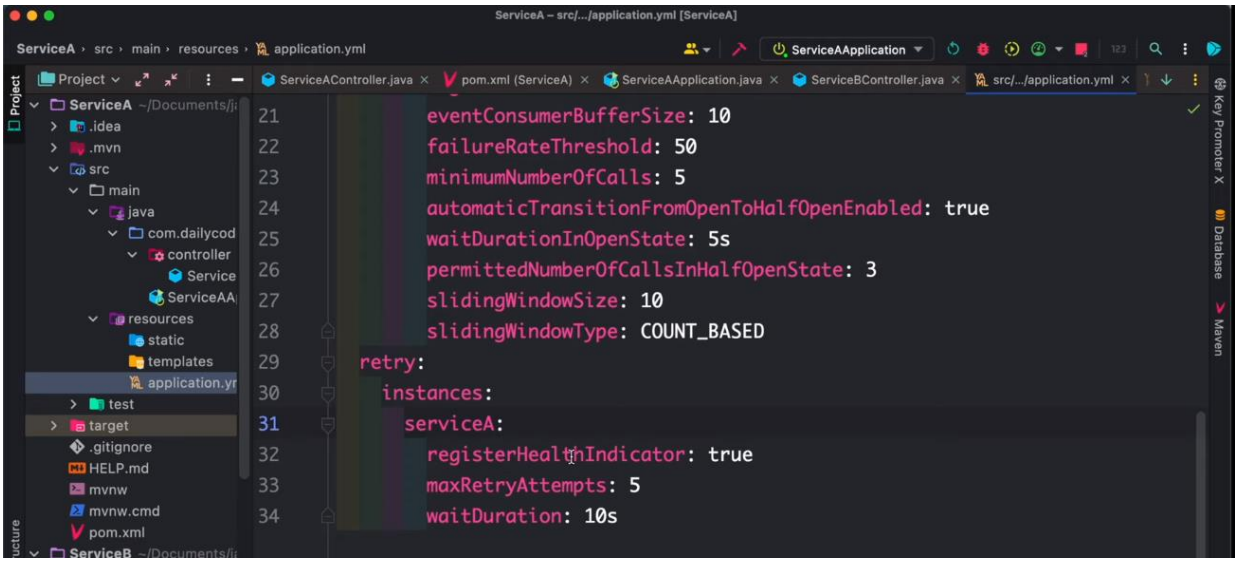


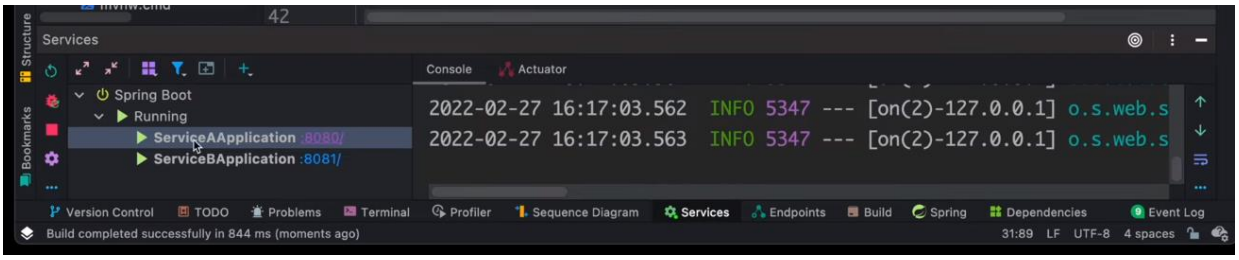
Next, let us now implement the Retry mechanism for our applications

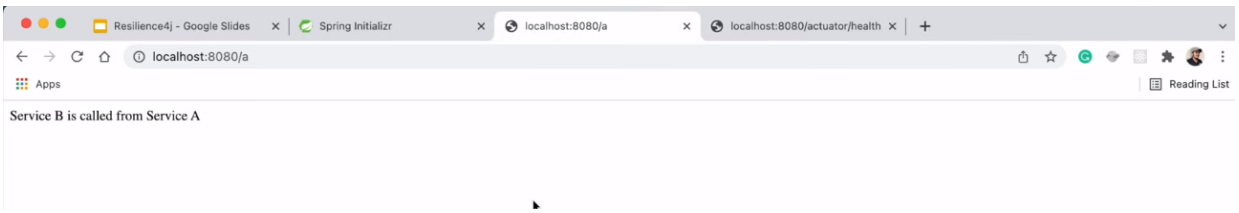


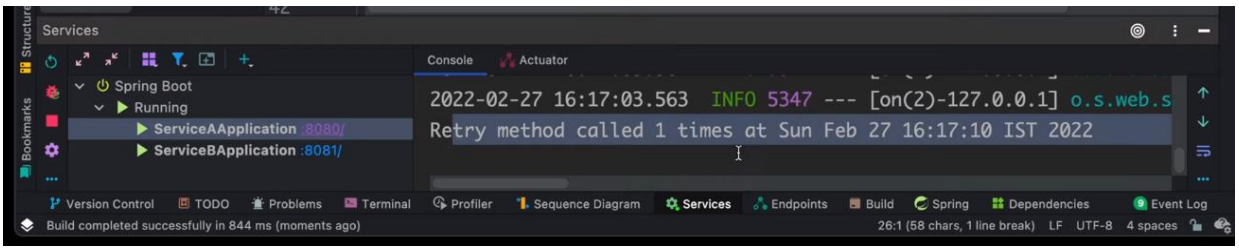
We can now add the configuration details for the Retry

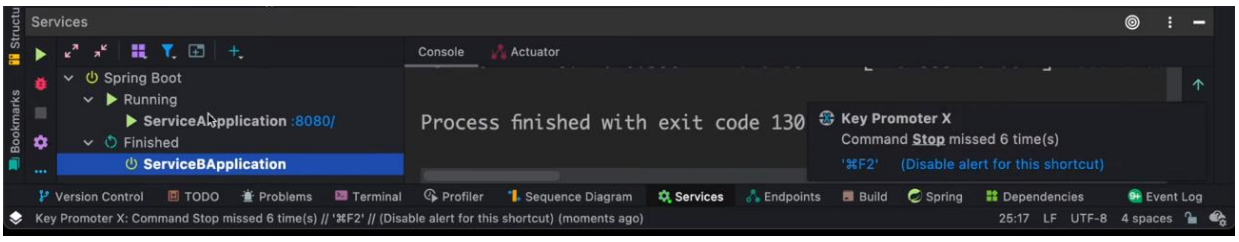




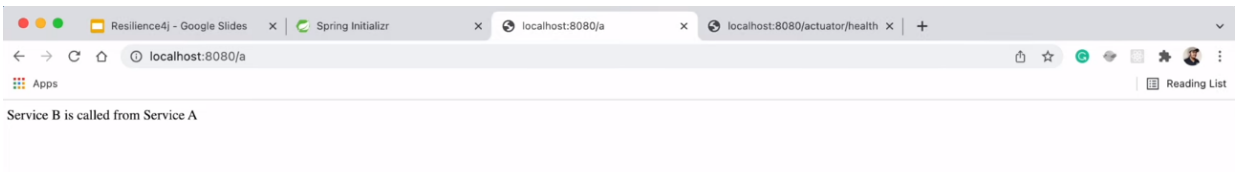


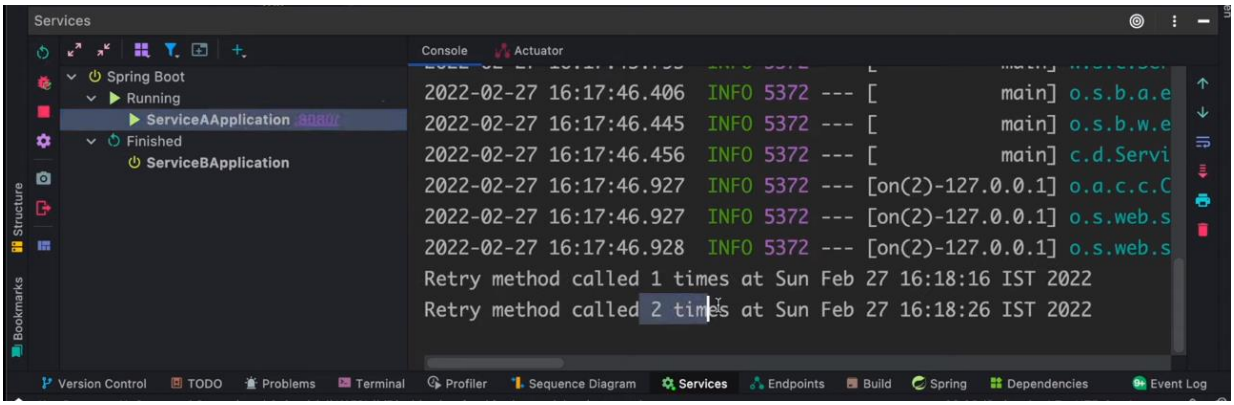


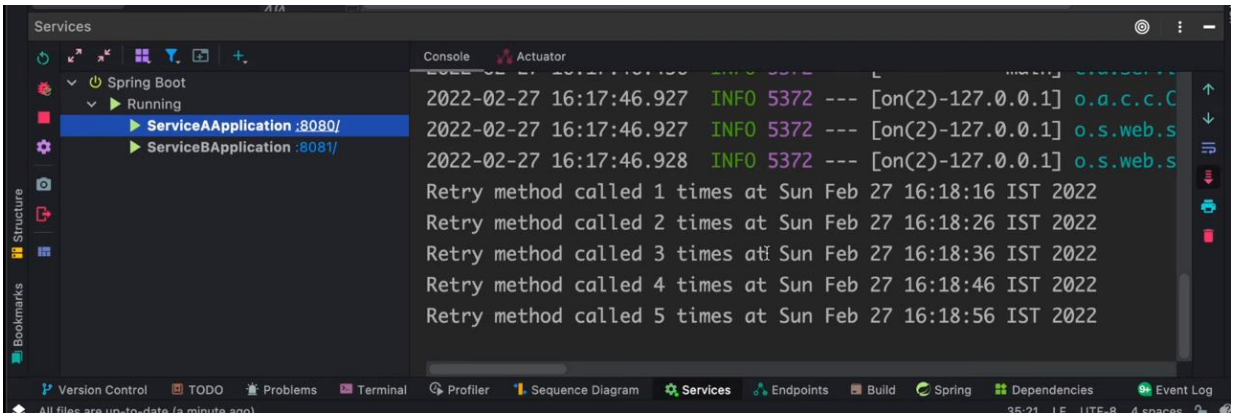


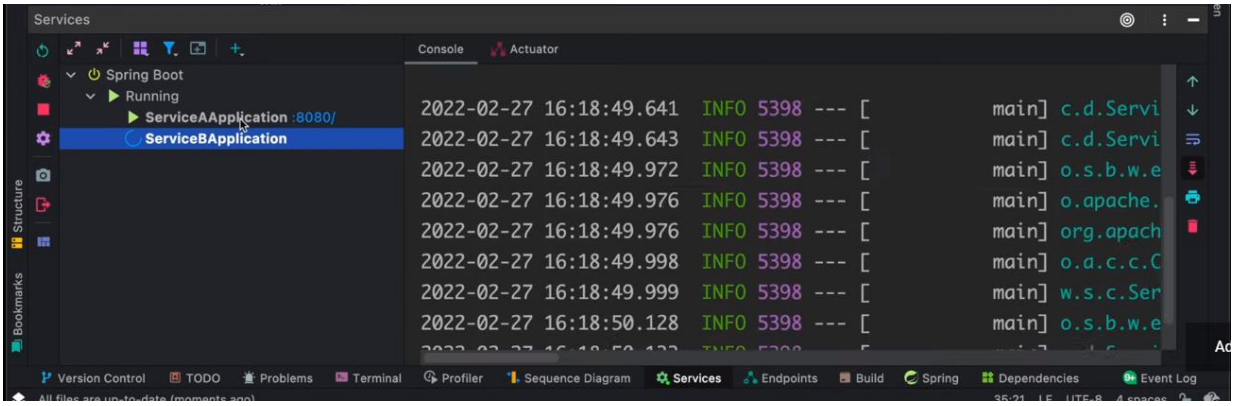


Stop service B now

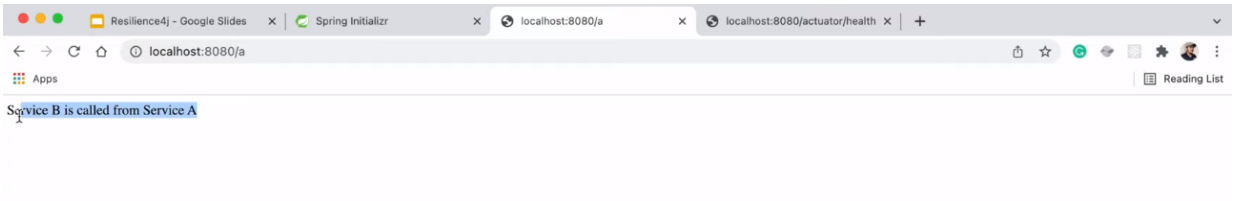




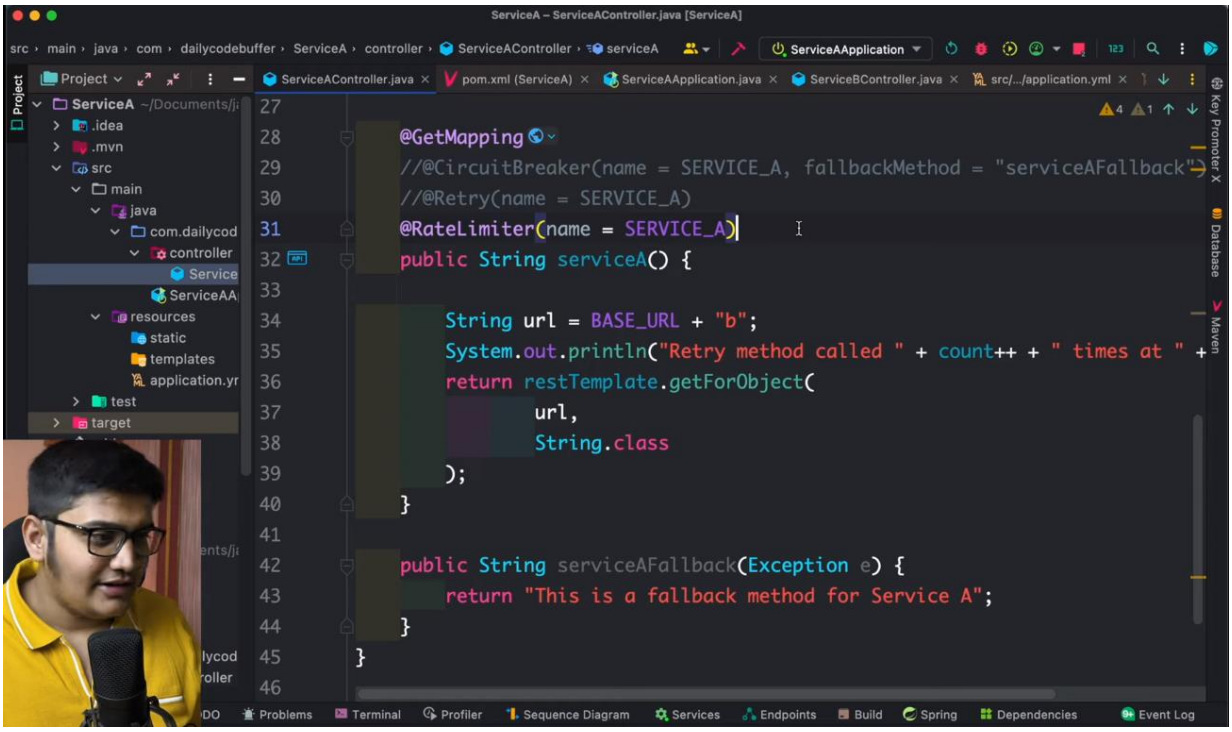




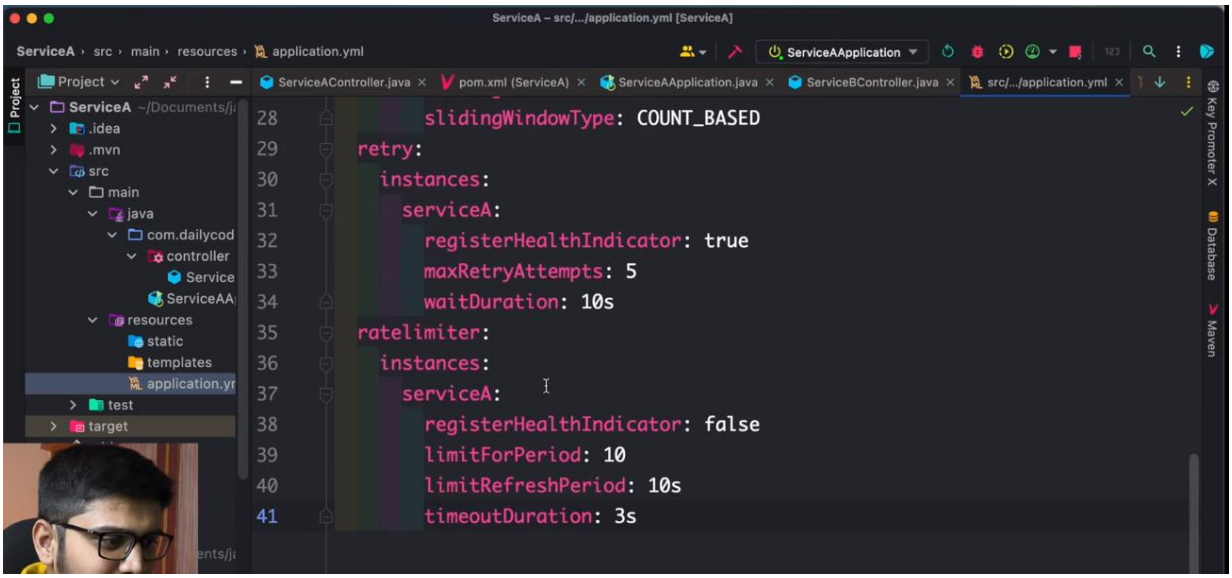
Start service B now



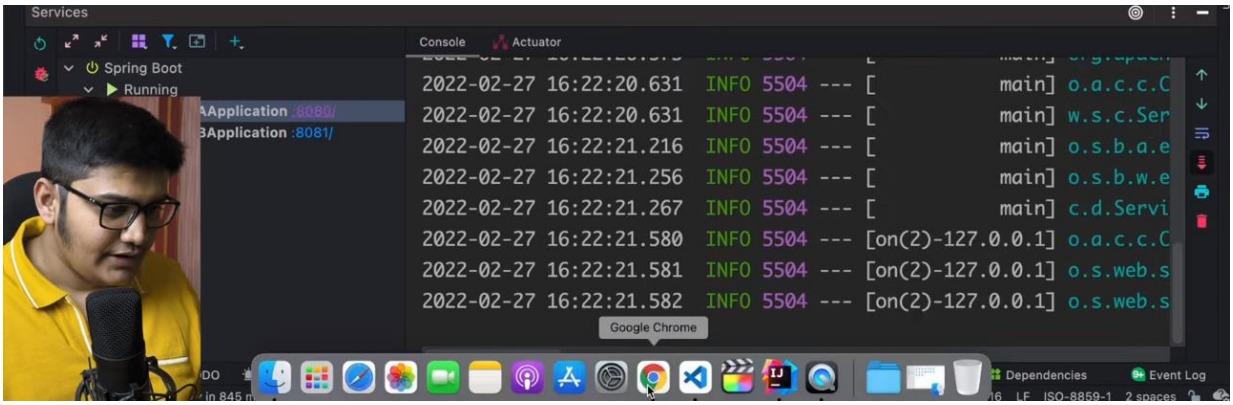
Service calls are now working again.

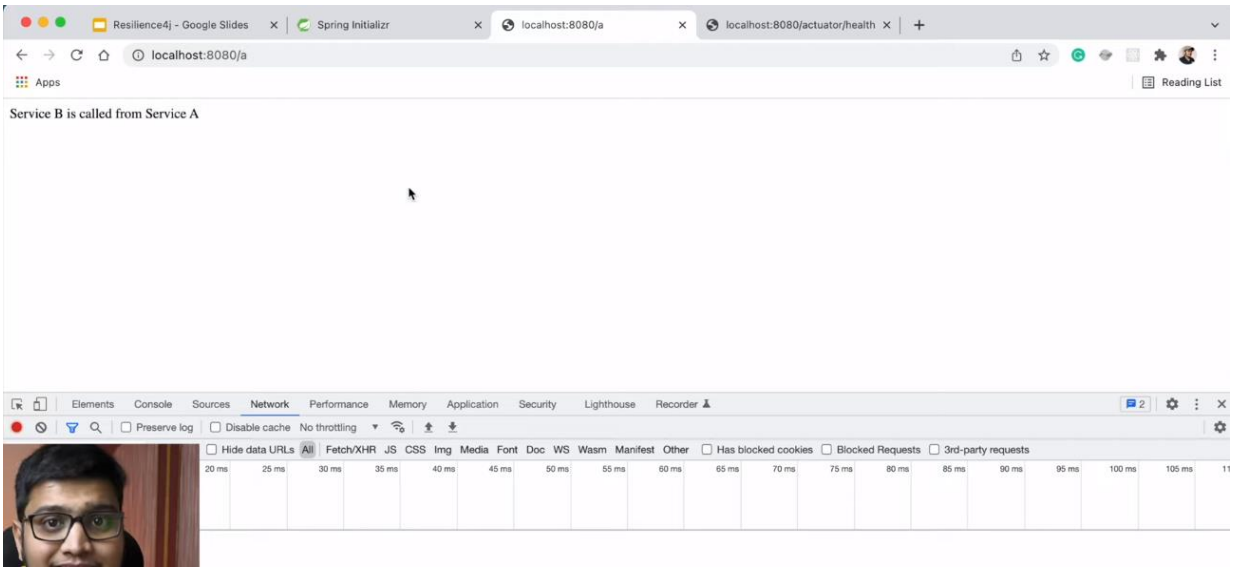


Next, let us implement the Rate Limiter using Resilience4j

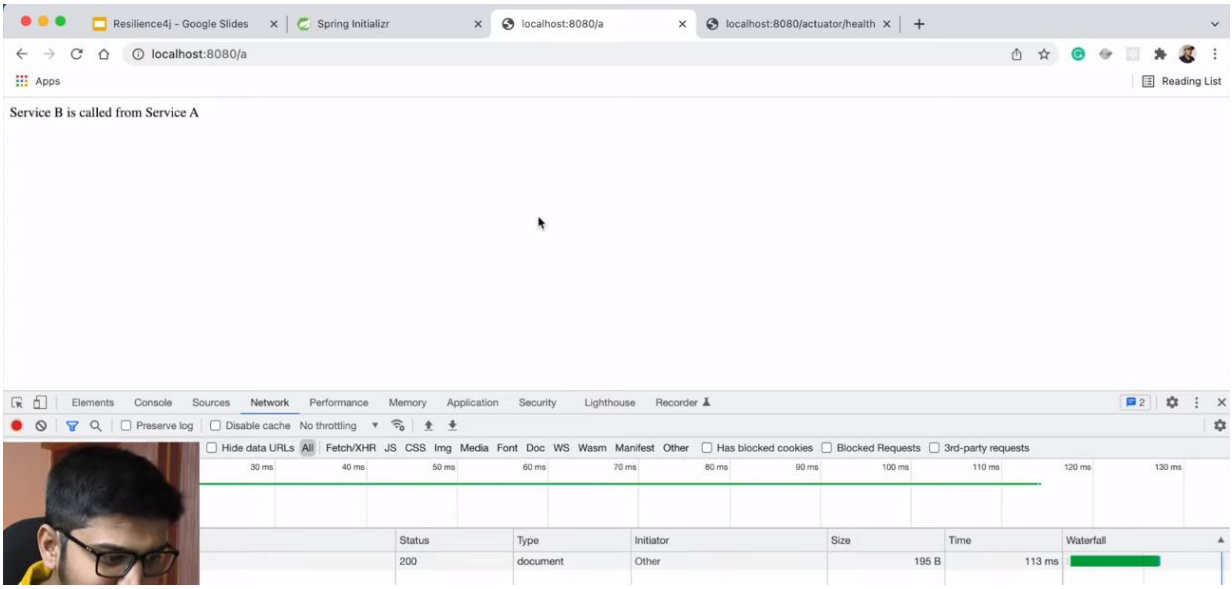


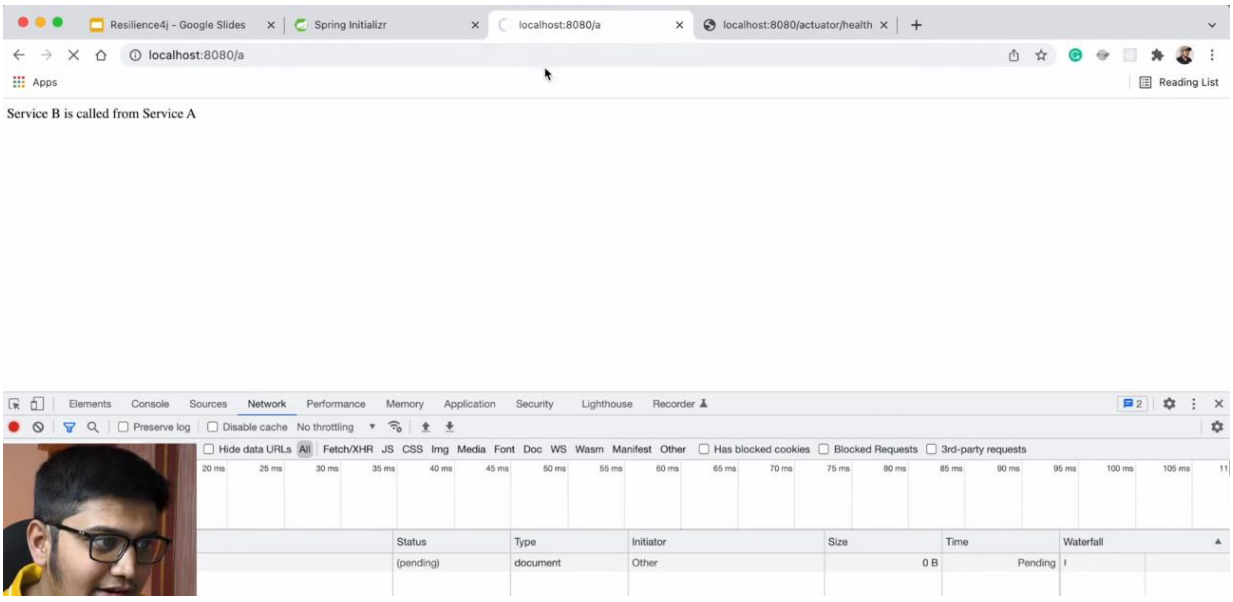
If you have multiple instances, you can define the default configuration too



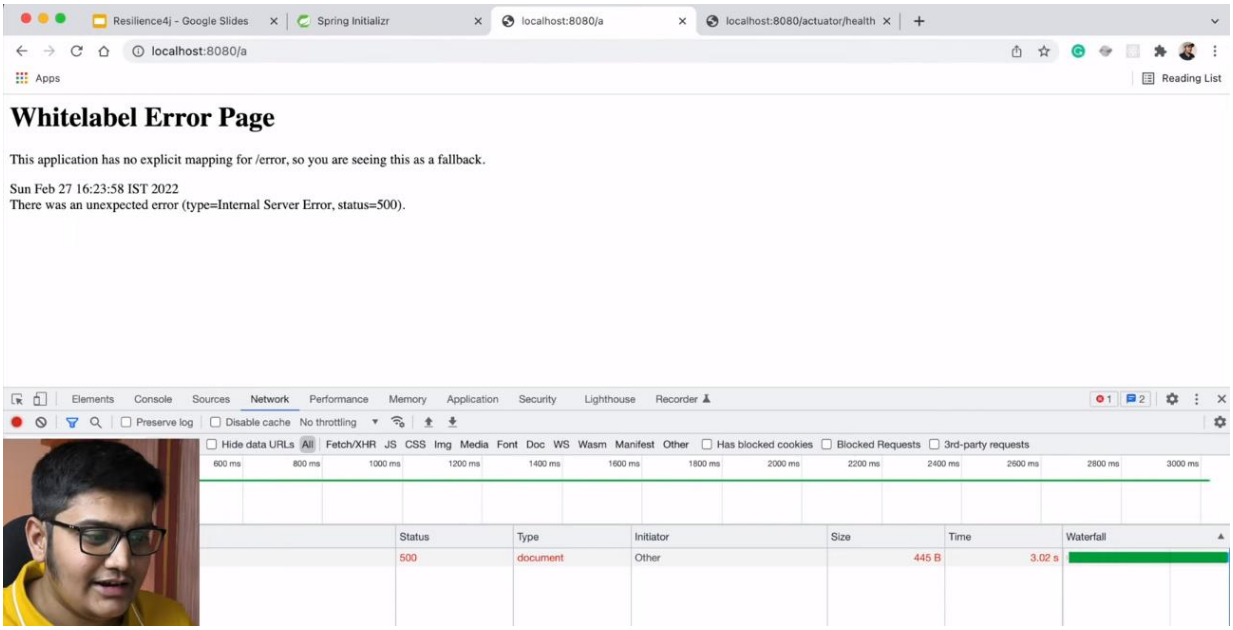


Only 10 requests are allowed within a 10s interval, a delay of 3s will occur if you try to exceed that limit

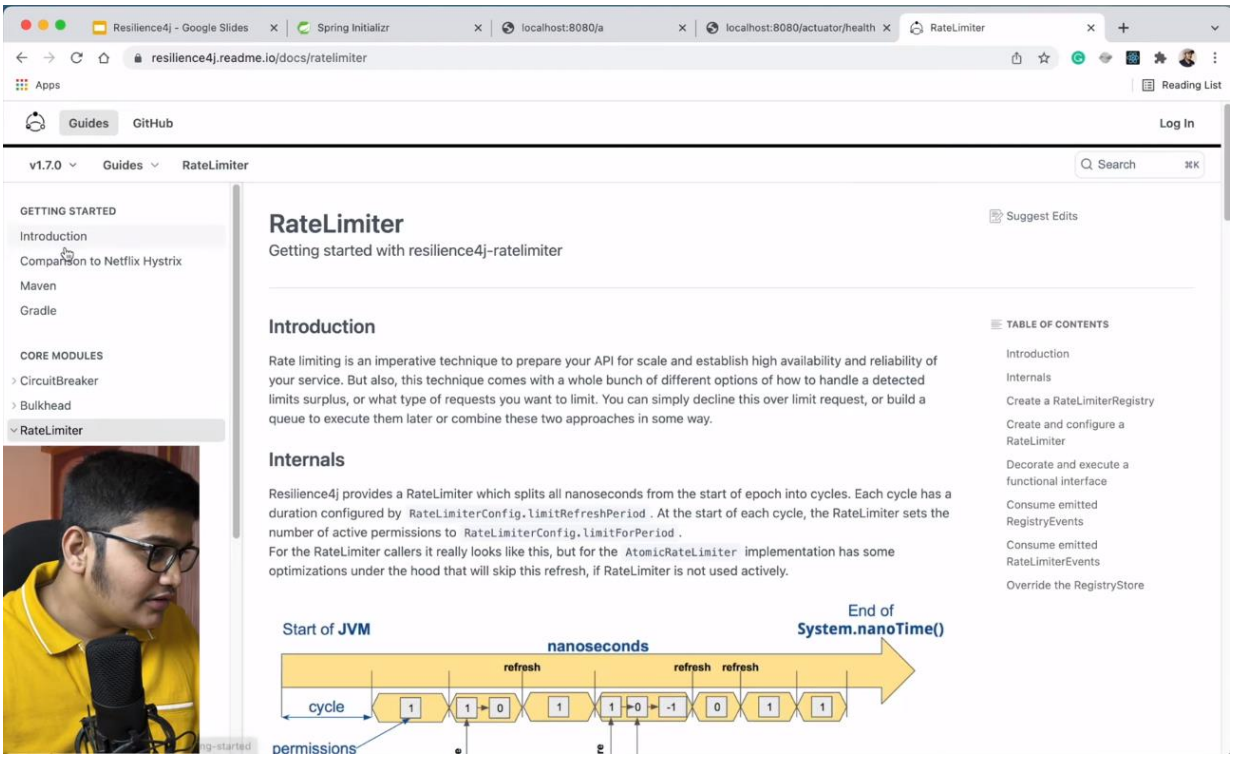




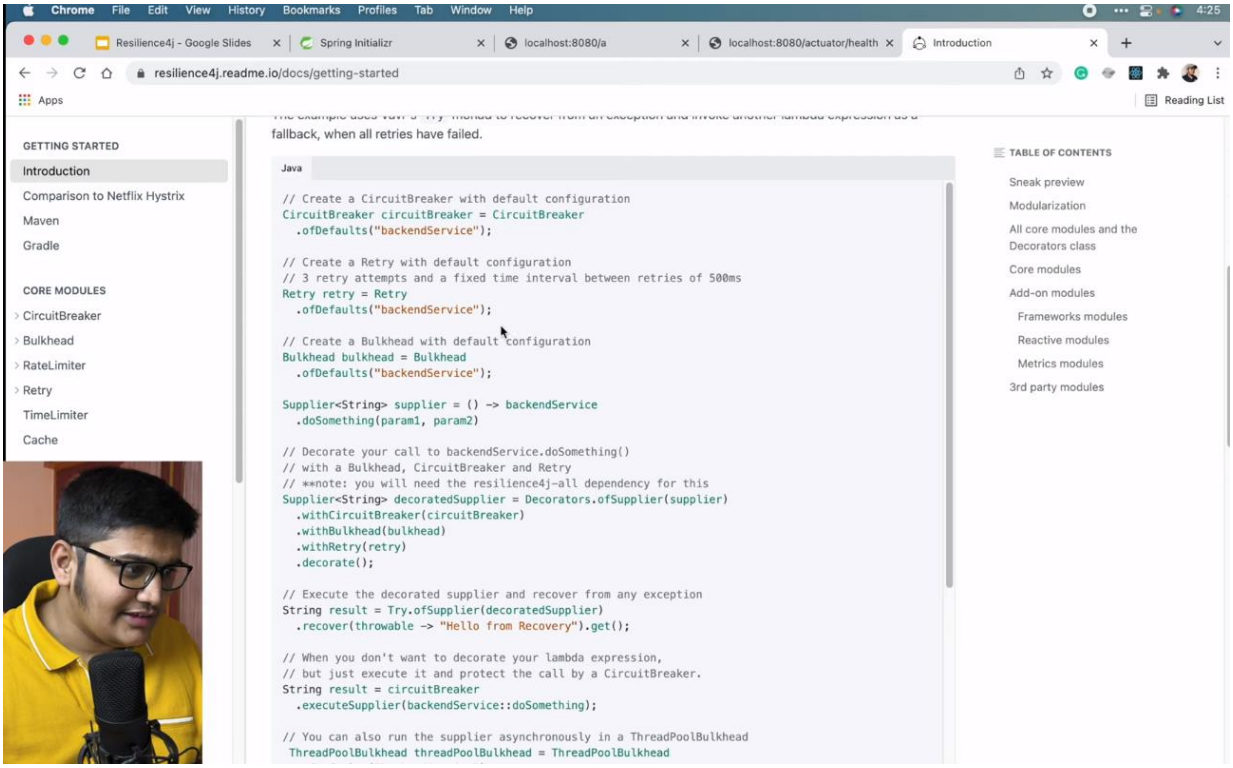
It is now waiting because we have sent 10 requests within the 10s interval



After 3s, we will then get an error because we have not defined/used a fallback method for the rate limiter configuration



https://www.resilience4j.readme.io/docs/getting-started



All the possible modules we can implement are given on the left pane for spring-boot, reactor, etc