Restful Spring based provider documentation – Part 2

Adv. & Distributed Programming Paradigms

CSC 3374 – 01

Main menu:

1- **Homework objective:**

In this application the consumer will be able to take control of the remote system asynchronously by:

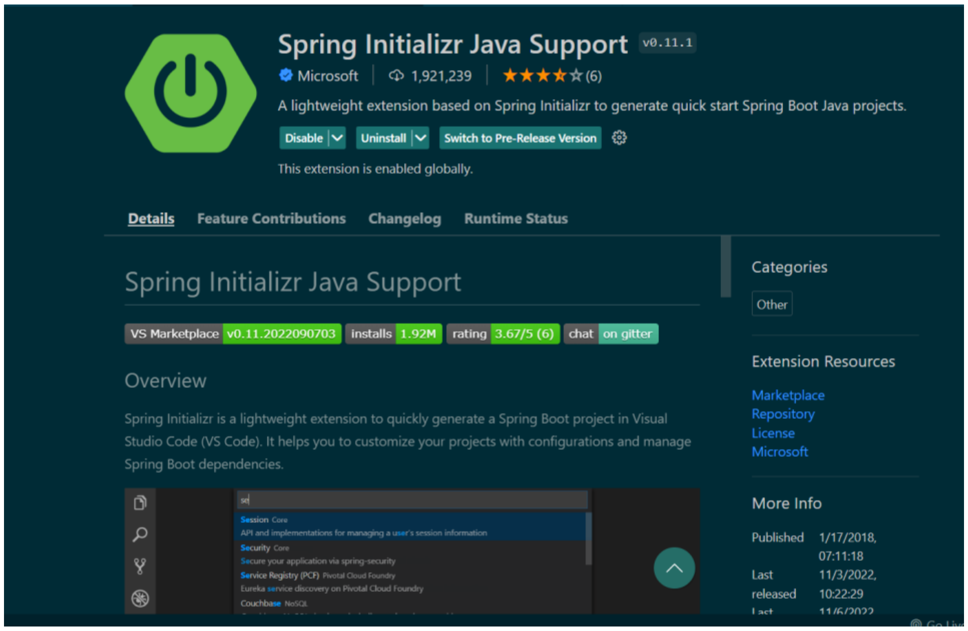
- Reboot the remote system.
- Take screenshot of the remote system.
- Get the list of running processes in the remote system.

2- **Development approach:**

By following the first code approach:

Service provider:

First, we initialized our spring boot project by initializing the necessary dependencies.



Spring Initializr Java Support v0.11.1

Microsoft | 1,921,239 | ★★★★☆ (6)

A lightweight extension based on Spring Initializr to generate quick start Spring Boot Java projects.

This extension is enabled globally.

Spring Initializr Java Support

VS Marketplace v0.11.2022090703 | installs 1.92M | rating 3.67/5 (6) | chat on gitter

Overview

Spring Initializr is a lightweight extension to quickly generate a Spring Boot project in Visual Studio Code (VS Code). It helps you to customize your projects with configurations and manage Spring Boot dependencies.

```
se
Session Core
API and implementations for managing a user's session information
Security Core
Secure your application via spring-security
Service Registry (PCF) Pivotal Cloud Foundry
Eureka service discovery on Pivotal Cloud Foundry
Couchbase NoSQL
```

Categories

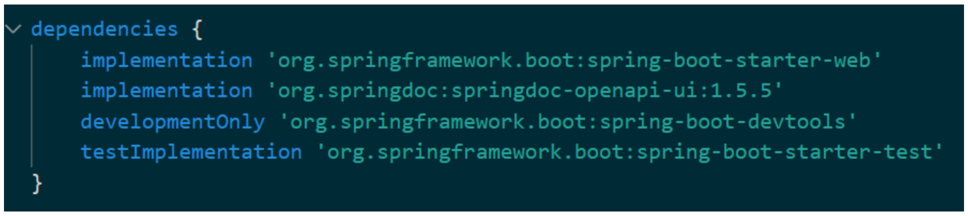
Other

Extension Resources

Marketplace
Repository
License
Microsoft

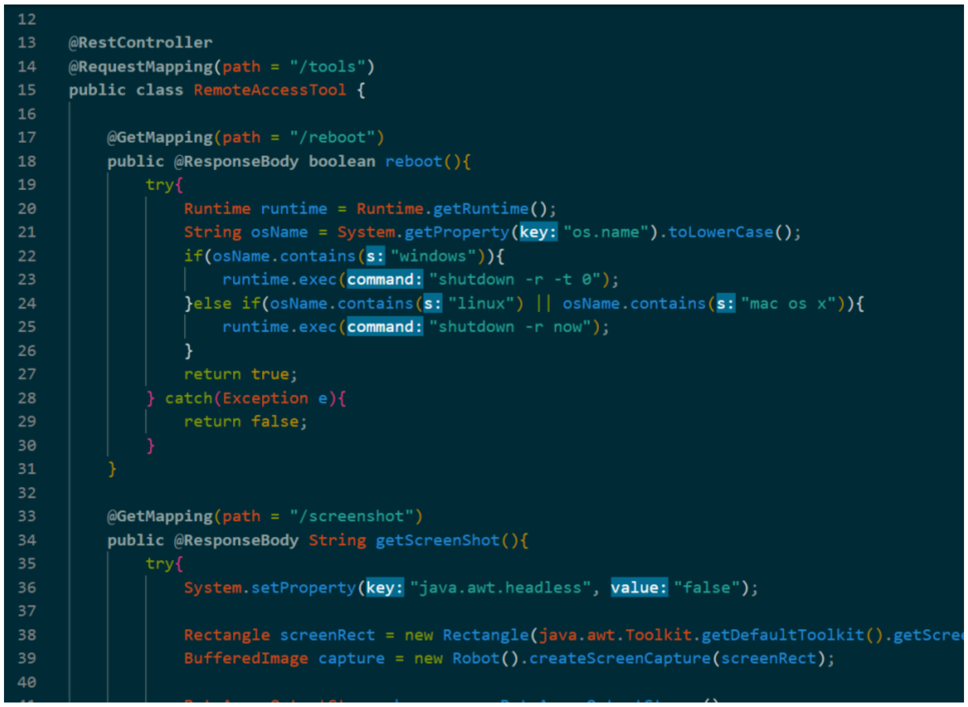
More Info

Published	1/17/2018, 07:11:18
Last released	11/3/2022, 10:22:29



```
dependencies {
    implementation 'org.springframework.boot:spring-boot-starter-web'
    implementation 'org.springdoc:springdoc-openapi-ui:1.5.5'
    developmentOnly 'org.springframework.boot:spring-boot-devtools'
    testImplementation 'org.springframework.boot:spring-boot-starter-test'
}
```

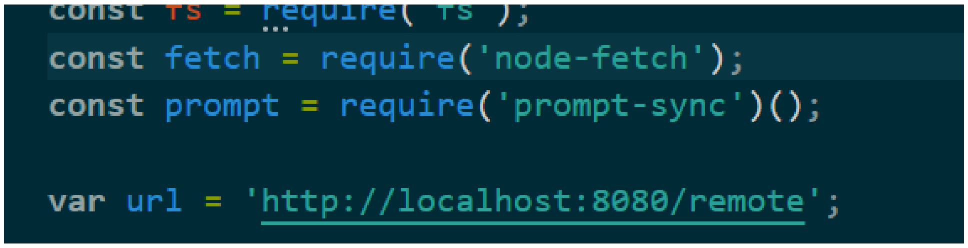
After that, we implemented our business implementation of the service that we provide (reboot, screenshot, list of running processes)methods. Then, we mark our “RemoteAccessTool” class as web service using “@RestController” annotation of spring boot. To map HTTP requests to the REST controller class, we additionally utilized the "@RequestMapping" annotation. Eventually, To ensure that HTTP GET requests are mapped to the appropriate methods, use the "@GetMapping" annotation.



```

@RestController
@RequestMapping(path = "/tools")
public class RemoteAccessTool {

    @GetMapping(path = "/reboot")
    public @ResponseBody boolean reboot(){
        try{
            Runtime runtime = Runtime.getRuntime();
            String osName = System.getProperty(key: "os.name").toLowerCase();
            if(osName.contains(s: "windows")){
                runtime.exec(command: "shutdown -r -t 0");
            }else if(osName.contains(s: "linux") || osName.contains(s: "mac os x")){
                runtime.exec(command: "shutdown -r now");
            }
            return true;
        } catch(Exception e){
            return false;
        }
    }

    @GetMapping(path = "/screenshot")
    public @ResponseBody String getScreenShot(){
        try{
            System.setProperty(key: "java.awt.headless", value: "false");

            Rectangle screenRect = new Rectangle(java.awt.Toolkit.getDefaultToolkit().getScre
            BufferedImage capture = new Robot().createScreenCapture(screenRect);

```

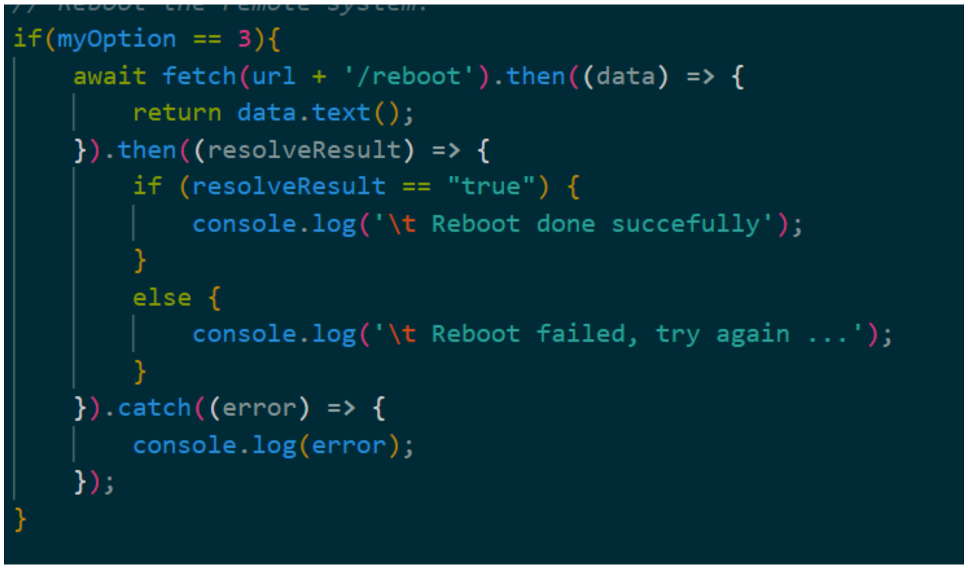
Consumer:

We created the promise-based consumer on this side, who will use the service being provided asynchronously. The service's URL, which also specifies the path of the resource we wish to fetch, was initially declared.



```
const fs = require('fs');
const fetch = require('node-fetch');
const prompt = require('prompt-sync')();

var url = 'http://localhost:8080/remote';
```

The response to that request is the promise returned by the get() method, which we then called. After retrieving the answer, we utilized the method.text(), which also yields a promise that resolves to the body's contents.



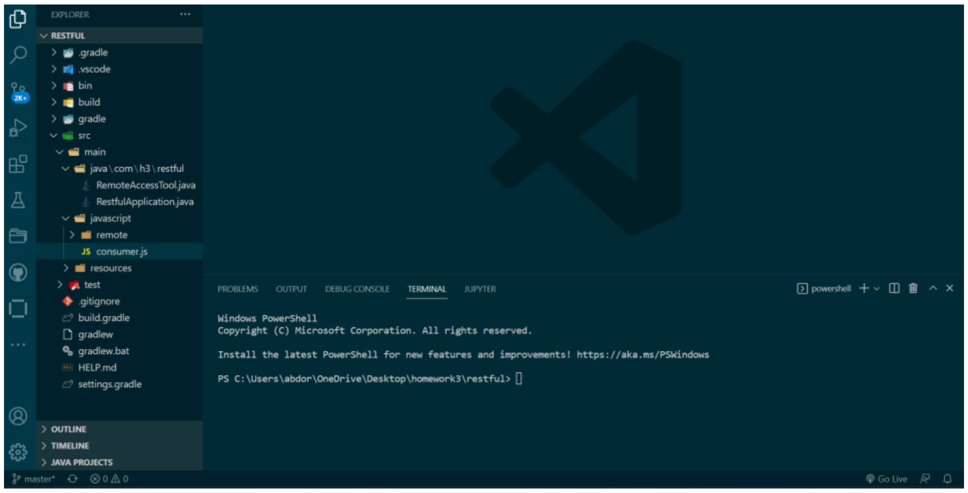
```
if(myOption == 3){
    await fetch(url + '/reboot').then((data) => {
        return data.text();
    }).then((resolveResult) => {
        if (resolveResult == "true") {
            console.log('\t Reboot done succefully');
        }
        else {
            console.log('\t Reboot failed, try again ...');
        }
    }).catch((error) => {
        console.log(error);
    });
}
```

3- **<u>Steps to run:</u>**

Provider:

To start the service, you should follow the below steps:

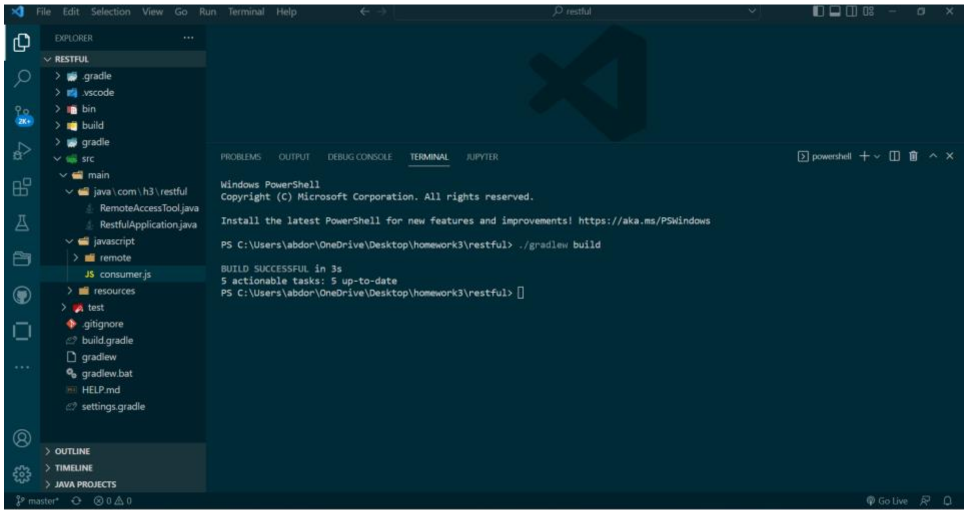
- Open your command line terminal:

- 

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\abdor\OneDrive\Desktop\homework3\restful>
```

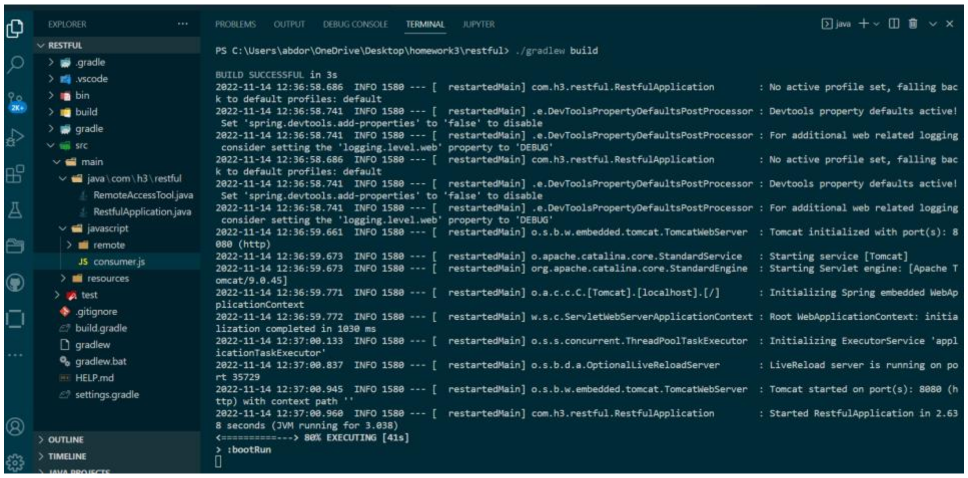
- Be sure that you're under your gradle project

- Build project: ./gradlew build



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\abdor\OneDrive\Desktop\homework3\restful> ./gradlew build

BUILD SUCCESSFUL in 3s
5 actionable tasks: 5 up-to-date
PS C:\Users\abdor\OneDrive\Desktop\homework3\restful>
```

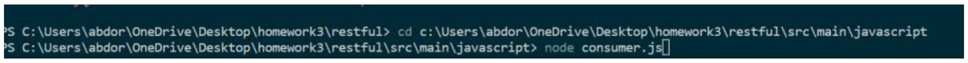
- Run the project: ./gradlew bootRun



```
PS C:\Users\abdor\OneDrive\Desktop\homework3\restful> ./gradlew build

BUILD SUCCESSFUL in 3s
2022-11-14 12:36:58.686  INFO 1580 --- [  restartedMain] com.h3.restful.RestfulApplication        : No active profile set, falling bac
k to default profiles: default
2022-11-14 12:36:58.741  INFO 1580 --- [  restartedMain] .e.DevToolsPropertyDefaultsPostProcessor : Devtools property defaults active!
 Set 'spring.devtools.add-properties' to 'false' to disable
2022-11-14 12:36:58.741  INFO 1580 --- [  restartedMain] .e.DevToolsPropertyDefaultsPostProcessor : For additional web related logging
 consider setting the 'logging.level.web' property to 'DEBUG'
2022-11-14 12:36:58.686  INFO 1580 --- [  restartedMain] com.h3.restful.RestfulApplication        : No active profile set, falling bac
k to default profiles: default
2022-11-14 12:36:58.741  INFO 1580 --- [  restartedMain] .e.DevToolsPropertyDefaultsPostProcessor : Devtools property defaults active!
 Set 'spring.devtools.add-properties' to 'false' to disable
2022-11-14 12:36:58.741  INFO 1580 --- [  restartedMain] .e.DevToolsPropertyDefaultsPostProcessor : For additional web related logging
 consider setting the 'logging.level.web' property to 'DEBUG'
2022-11-14 12:36:59.661  INFO 1580 --- [  restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8
080 (http)
2022-11-14 12:36:59.673  INFO 1580 --- [  restartedMain] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2022-11-14 12:36:59.673  INFO 1580 --- [  restartedMain] org.apache.catalina.core.StandardEngine  : Starting Servlet engine: [Apache T
omcat/9.0.45]
2022-11-14 12:36:59.771  INFO 1580 --- [  restartedMain] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebAp
plicationContext
2022-11-14 12:36:59.772  INFO 1580 --- [  restartedMain] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initia
lization completed in 1030 ms
2022-11-14 12:37:00.133  INFO 1580 --- [  restartedMain] o.s.s.concurrent.ThreadPoolTaskExecutor  : Initializing ExecutorService 'appl
icationTaskExecutor'
2022-11-14 12:37:00.837  INFO 1580 --- [  restartedMain] o.s.b.d.a.OptionalLiveReloadServer       : LiveReload server is running on po
rt 35729
2022-11-14 12:37:00.945  INFO 1580 --- [  restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8080 (h
ttp) with context path ''
2022-11-14 12:37:00.960  INFO 1580 --- [  restartedMain] com.h3.restful.RestfulApplication        : Started RestfulApplication in 2.63
8 seconds (JVM running for 3.038)
<==========---> 80% EXECUTING [41s]
> :bootRun
```

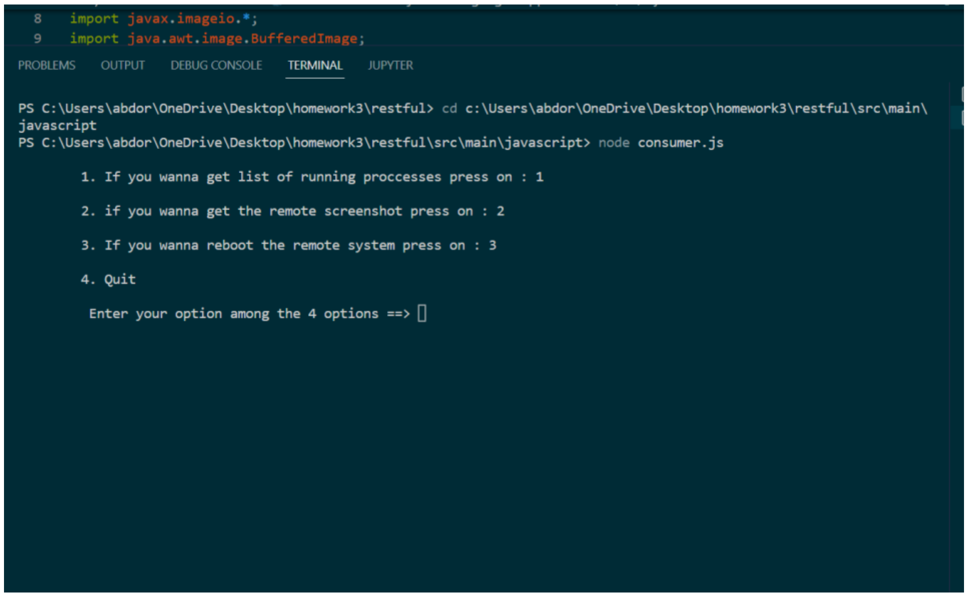
Consumer:

- Open your command line terminal
- Change your directory where you have you js consumer file
- Run: node consumer.js



```
PS C:\Users\abdor\OneDrive\Desktop\homework3\restful> cd c:\Users\abdor\OneDrive\Desktop\homework3\restful\src\main\javascript
PS C:\Users\abdor\OneDrive\Desktop\homework3\restful\src\main\javascript> node consumer.js
```

- Once the application is running a menu will display of 4 options :



```
import javax.imageio.*;
import java.awt.image.BufferedImage;
```

```
PS C:\Users\abdor\OneDrive\Desktop\homework3\restful> cd c:\Users\abdor\OneDrive\Desktop\homework3\restful\src\main\
javascript
PS C:\Users\abdor\OneDrive\Desktop\homework3\restful\src\main\javascript> node consumer.js

        1. If you wanna get list of running proccesses press on : 1

        2. if you wanna get the remote screenshot press on : 2

        3. If you wanna reboot the remote system press on : 3

        4. Quit

         Enter your option among the 4 options ==> 
```

Then the client choose the option needed.

4- Technologies used:

Service definition language:

The OpenAPI Specification (OAS) defines a standard, language-agnostic interface to RESTful APIs which allows both humans and computers to discover and understand the capabilities of the service without access to source code, documentation, or through network traffic inspection. When properly defined, a consumer can understand and interact with the remote service with a minimal amount of implementation logic. (swagger docs)

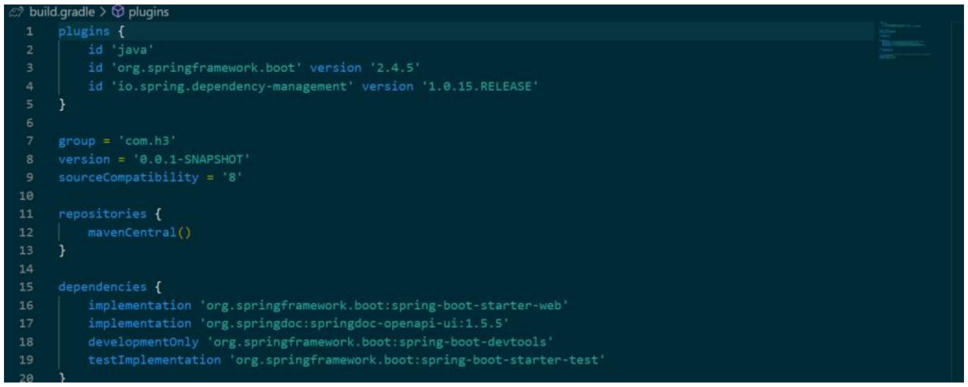
Protocol:

Hypertext Transfer Protocol (HTTP) is an application-layer protocol for transmitting hypermedia documents.

Provider:

- Java version 17.0.5

- Spring boot: (Spring Framework) is a popular, open source, framework for creating standalone, production-grade applications that run on the Java Virtual Machine (JVM).

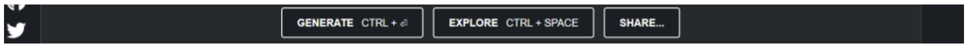


build.gradle > plugins

```
plugins {
    id 'java'
    id 'org.springframework.boot' version '2.4.5'
    id 'io.spring.dependency-management' version '1.0.15.RELEASE'
}

group = 'com.h3'
version = '0.0.1-SNAPSHOT'
sourceCompatibility = '8'

repositories {
    mavenCentral()
}

dependencies {
    implementation 'org.springframework.boot:spring-boot-starter-web'
    implementation 'org.springdoc:springdoc-openapi-ui:1.5.5'
    developmentOnly 'org.springframework.boot:spring-boot-devtools'
    testImplementation 'org.springframework.boot:spring-boot-starter-test'
}
```

Additional tools:

- Spring initializer: in order to generate our spring boot application we can use: https://start.spring.io



In this application I used spring intializr extension in vscode:

- Graddle: a build automation tool that is designed to be flexible enough to build almost any type of software

__Consumer:__

- __Javascript__

```
}
else {
```

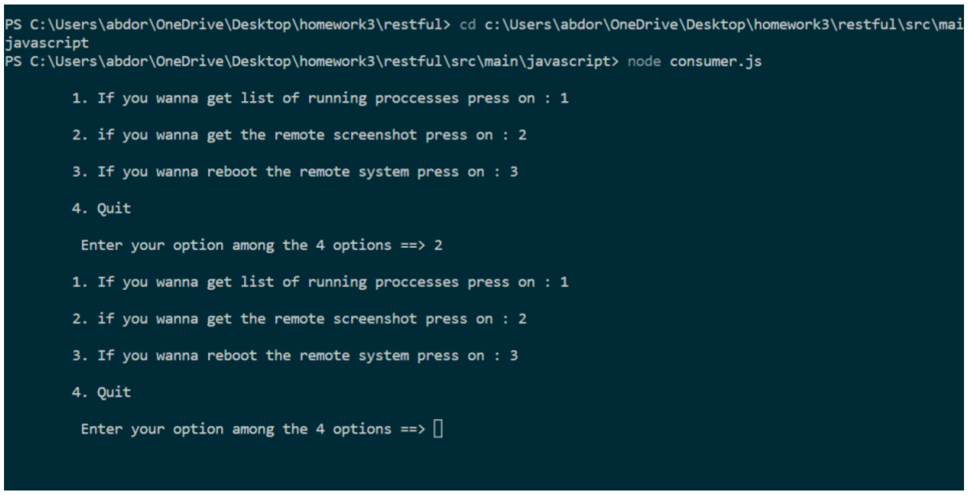
- __Nodejs:__ is runtime environment to execute javascript
-

__Fetch API:__ The Fetch API provides an interface for fetching resources (including across the network). It will seem familiar to anyone who has used __`XMLHttpRequest`__, but the new API provides a more powerful and flexible feature set. (developer modzilla)

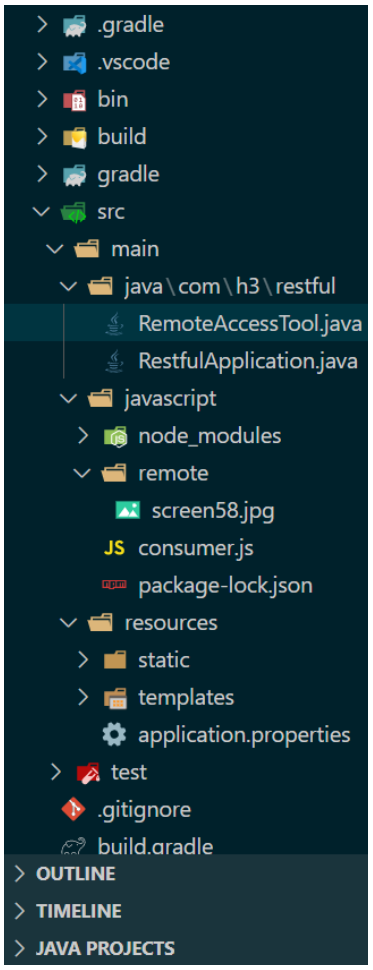
5- __Screenshots:__

__Get screenshots from the remote system:__

I assigned a random number to the screenshot name, so each time we take a screenshot a random number is attached at the end of image name; for example: “”screen58”

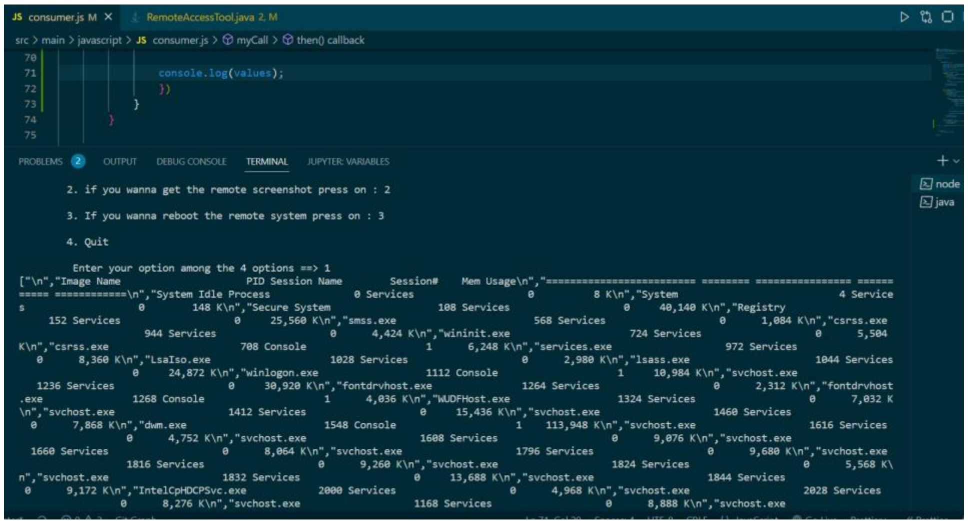


```
PS C:\Users\abdor\OneDrive\Desktop\homework3\restful> cd c:\Users\abdor\OneDrive\Desktop\homework3\restful\src\mai
javascript
PS C:\Users\abdor\OneDrive\Desktop\homework3\restful\src\main\javascript> node consumer.js

        1. If you wanna get list of running proccesses press on : 1

        2. if you wanna get the remote screenshot press on : 2

        3. If you wanna reboot the remote system press on : 3

        4. Quit

         Enter your option among the 4 options ==> 2

        1. If you wanna get list of running proccesses press on : 1

        2. if you wanna get the remote screenshot press on : 2

        3. If you wanna reboot the remote system press on : 3

        4. Quit

         Enter your option among the 4 options ==> 
```



```
> .gradle
> .vscode
> bin
> build
> gradle
v src
  v main
    v java\com\h3\restful
        RemoteAccessTool.java
        RestfulApplication.java
    v javascript
      > node_modules
      v remote
          screen58.jpg
        consumer.js
        package-lock.json
    v resources
      > static
      > templates
        application.properties
  > test
  .gitignore
  build.gradle
> OUTLINE
> TIMELINE
> JAVA PROJECTS
```

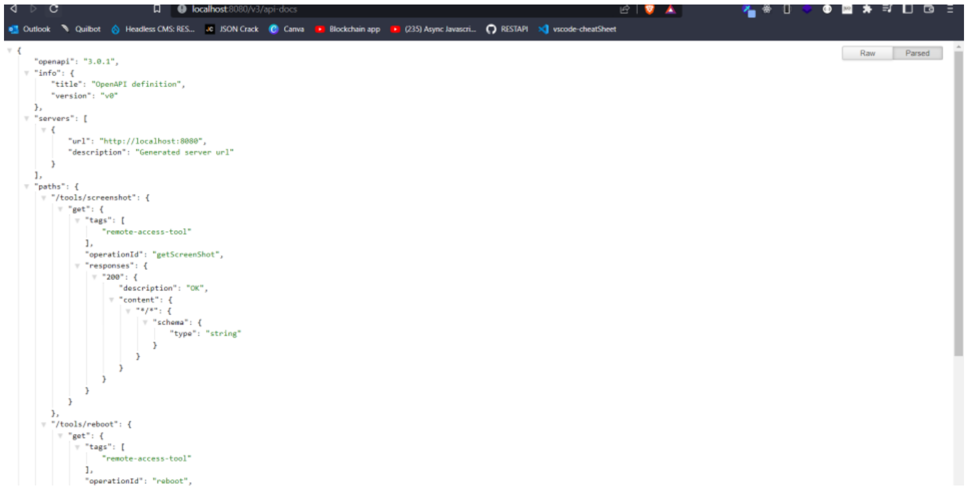
Get list of running processes:


consumer.js M
RemoteAccessTool.java 2, M
src > main > javascript > consumer.js > myCall > then() callback
PROBLEMS 2
OUTPUT
DEBUG CONSOLE
TERMINAL
JUPYTER: VARIABLES
node
java

```
                        console.log(values);
                        })
                    }
                }
```

```
        2. if you wanna get the remote screenshot press on : 2

        3. If you wanna reboot the remote system press on : 3

        4. Quit

         Enter your option among the 4 options ==> 1
["\n","Image Name                     PID Session Name        Session#    Mem Usage\n","========================= ======== ================ ======
===== ============\n","System Idle Process              0 Services                   0          8 K\n","System                          4 Service
s                     0        148 K\n","Secure System                    108 Services                   0     40,140 K\n","Registry
     152 Services                   0     25,560 K\n","smss.exe                         568 Services                   0      1,084 K\n","csrss.exe
                   944 Services                   0      4,424 K\n","wininit.exe                      724 Services                   0      5,504
K\n","csrss.exe                        708 Console                    1      6,248 K\n","services.exe                     972 Services
   0      8,360 K\n","LsaIso.exe                      1028 Services                   0      2,980 K\n","lsass.exe                       1044 Services
                   0     24,872 K\n","winlogon.exe                    1112 Console                    1     10,984 K\n","svchost.exe
   1236 Services                   0     30,920 K\n","fontdrvhost.exe                 1264 Services                   0      2,312 K\n","fontdrvhost
.exe                  1268 Console                    1      4,036 K\n","WUDFHost.exe                    1324 Services                   0      7,032 K
\n","svchost.exe                     1412 Services                   0     15,436 K\n","svchost.exe                     1460 Services
  0      7,868 K\n","dwm.exe                         1548 Console                    1    113,948 K\n","svchost.exe                     1616 Services
                   0      4,752 K\n","svchost.exe                     1608 Services                   0      9,076 K\n","svchost.exe
  1660 Services                   0      8,064 K\n","svchost.exe                     1796 Services                   0      9,680 K\n","svchost.exe
                  1816 Services                   0      9,260 K\n","svchost.exe                     1824 Services                   0      5,568 K\
n","svchost.exe                     1832 Services                   0     13,688 K\n","svchost.exe                     1844 Services
 0      9,172 K\n","IntelCpHDCPSvc.exe             2000 Services                   0      4,968 K\n","svchost.exe                     2028 Services
                  0      8,276 K\n","svchost.exe                     1168 Services                   0      8,888 K\n","svchost.exe
```

Open api service:



```
{
    "openapi": "3.0.1",
    "info": {
        "title": "OpenAPI definition",
        "version": "v0"
    },
    "servers": [
        {
            "url": "http://localhost:8080",
            "description": "Generated server url"
        }
    ],
    "paths": {
        "/tools/screenshot": {
            "get": {
                "tags": [
                    "remote-access-tool"
                ],
                "operationId": "getScreenShot",
                "responses": {
                    "200": {
                        "description": "OK",
                        "content": {
                            "*/*": {
                                "schema": {
                                    "type": "string"
                                }
                            }
                        }
                    }
                }
            }
        },
        "/tools/reboot": {
            "get": {
                "tags": [
                    "remote-access-tool"
                ],
                "operationId": "reboot",
```