



Lab RSMortgage Spring Boot Person Service
→ REST Microservice

RSMortgage Spring Boot Person Service

Udemy Spring Cloud Course
Presented By
Binit Datta

Rolling Stone Technology

Formatted: February, 2017

Table of Content

1.0 - Introduction	5
1.1 – Create a new Spring Starter Project.....	6
1.2 – Fill initial values	7
1.3 – Do not choose anything	8
1.4 – Click Finish Now	29
1.5 – Let Spring Tool Suite Prepare the Project.....	30
1.6 – Verify the project identification in the pom.xml	31
1.7 – Add Parent project	31
1.8 – Add Properties	31
1.9– Add web and actuator Dependencies.....	32
1.10 – Add security and tomcat dependencies	32
1.11 – Add jpa orm and h2 db dependencies	32
1.12 – Add spring test dependencies	33
1.13 – Add Jackson databind dependencies	33
1.14 – Add HAL Browser dependencies	33
1.15 – Add Swagger dependencies.....	34
1.16 – Add MySQL Driver dependencies	34
1.17 – Add maven build section	34
1.18 –Add properties for application.properties resource directory	35
1.19 –Create application.yml under resource directory	35
1.20 –Create domain package	37
1.21 –Create api.rest package	38
1.22 –Create exception package.....	39
1.23 –Create dao.jpa package	40
1.24 –Create service package	41
1.25 –Create Person domain class in domain package.....	42
1.26 –Do the following for the Person domain class	43
1.27 –Create RESTAPIErrorInfo domain class	43
1.28 –Create HTTP400Exception class in the exception pckage.....	44
1.29 –Create HTTP404Exception class in the exception pckage.....	45
1.30 –Create PersonRepository interface in the dao.jpa package	46
1.31 –Create ServiceProperties class in the service package.....	47
1.32 –Create PersonServiceHealth class in the service package	48
1.33 –Create PersonServiceEvent class in the service package.....	49
1.34 –Create PersonServiceListener class in the service package	50
1.35 –Create PersonService class in the service package.....	51
1.36 –Create AbstractRestHandler class in the rest.api package	53
1.37 –Add PersonController in the rest.api package.....	55
1.38 –Add RestControllerAspect in the com.rollingstone package.....	58
1.39 –Add Application class in the com.rollingstone package.....	60
1.40 –Open Git Bash in project folder	61
1.41 –Run build	61

1.42 –Build successful	62
1.43 –Now Run	63
1.44 –Verify Database.....	64
1.45 –Verify Database Data.....	65
1.46 –Verify Data Retrieval API is working.....	66
1.47 –Verify Data AOP for @Before and @AfterReturning.....	67
1.47 –Verify Single Data Retrieval API is working	68
1.48 –Verify Not Found is working	69
1.49 –Verify Person Creation is working.....	70
1.49 –Verify Person Updation is working.....	72
1.50 –Verify Person Deletion is working	75
1.51 –Custom Health.....	76
1.54 – Metrics and Custom Metrics	77
1.55 – /env endpoint	78
1.56 – /beans endpoint	79
1.57 – /actuator endpoint.....	80
1.58 – /dump endpoint	82
1.59 – /logfile endpoint.....	83
1.60 –Conclusion	83

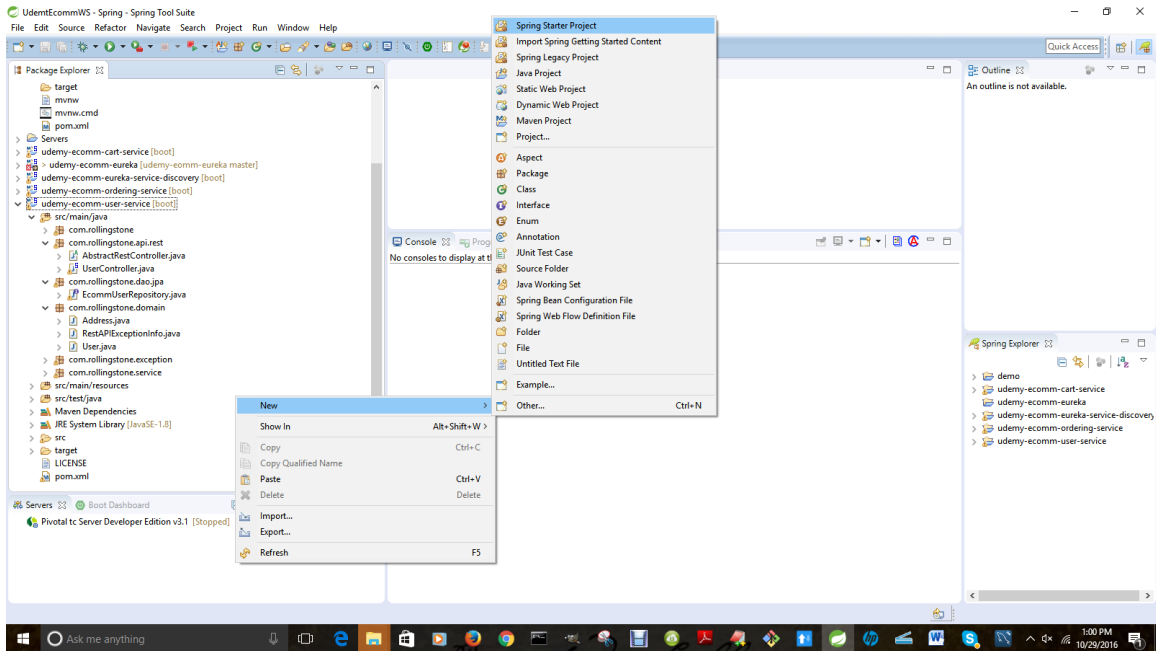
Chapter 1

Spring Boot Person Service Project Creation

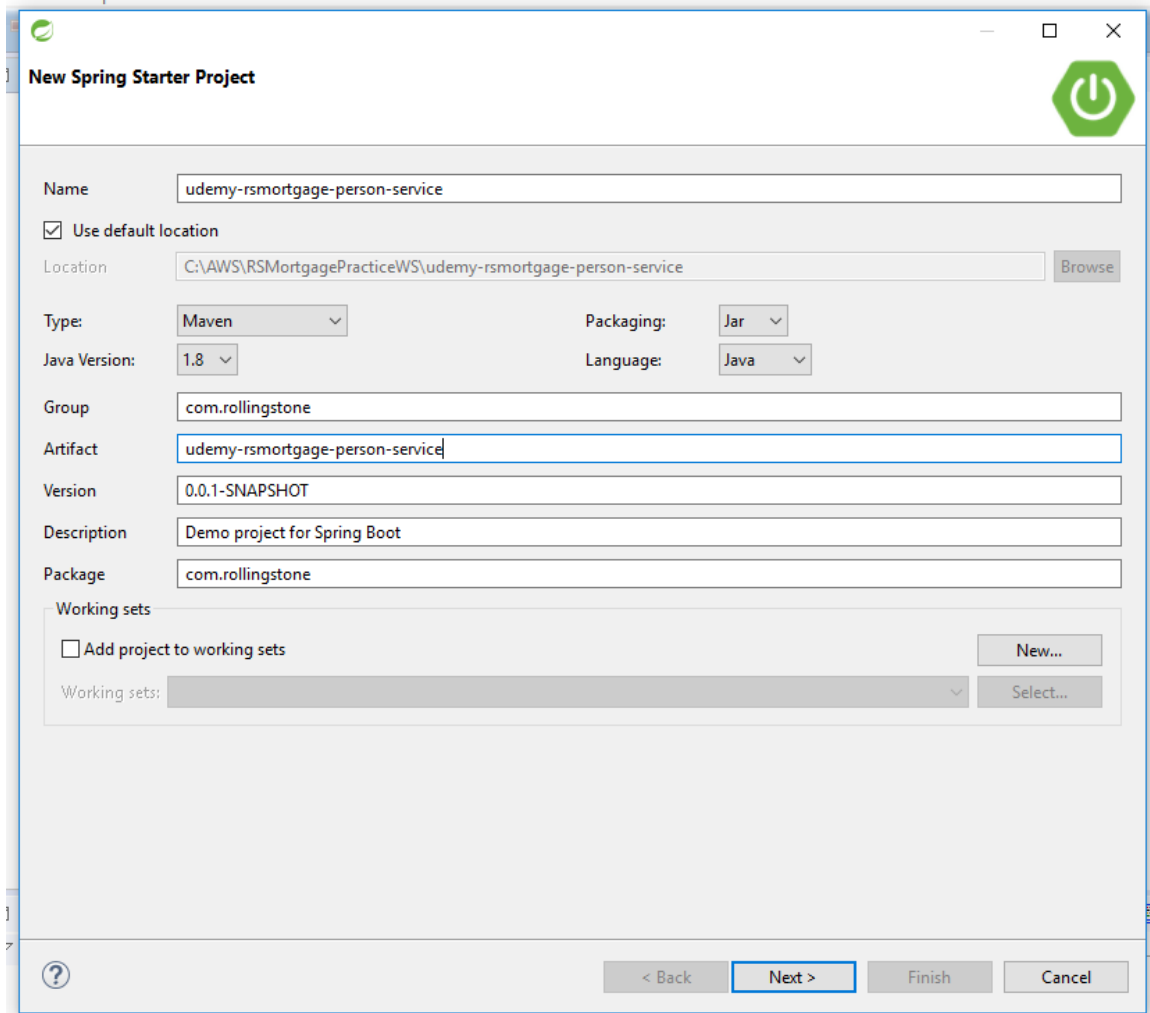
1.0 - Introduction

The purpose of this guide is to help walk through the process of creating a Spring Boot based Person Service REST Application so that the reader / learner becomes completely familiar with Spring Boot and its associated features such as actuator etc.

1.1 – Create a new Spring Starter Project



1.2 – Fill initial values



The image shows the 'New Spring Starter Project' dialog box in an IDE. The dialog is titled 'New Spring Starter Project' and features a green power button icon in the top right corner. The fields are filled with the following values:

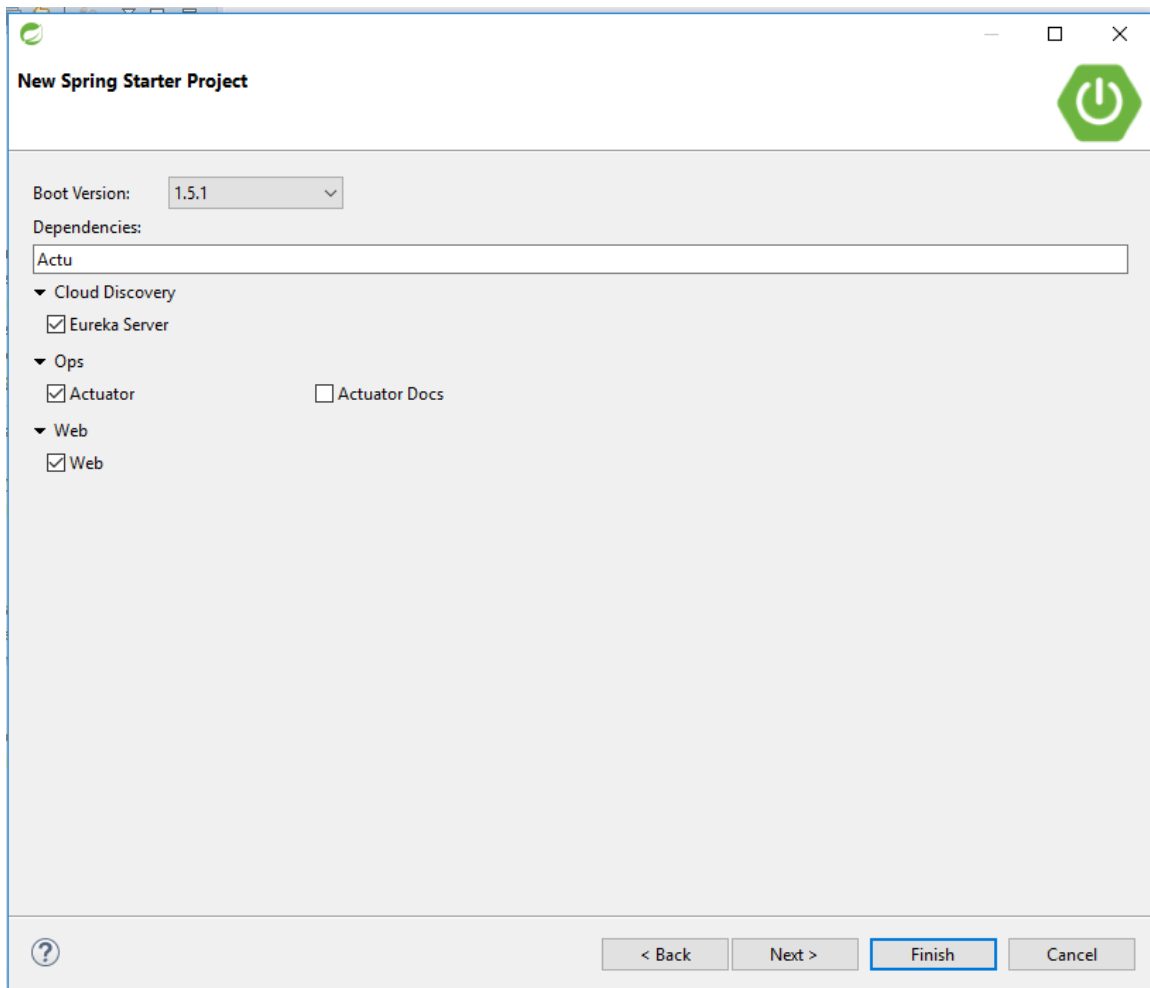
- Name: `udemy-rsmortgage-person-service`
- ☒ Use default location
- Location: `C:\AWS\RSMortgagePracticeWS\udemy-rsmortgage-person-service` (with a 'Browse' button)
- Type: `Maven` (dropdown)
- Packaging: `Jar` (dropdown)
- Java Version: `1.8` (dropdown)
- Language: `Java` (dropdown)
- Group: `com.rollingstone`
- Artifact: `udemy-rsmortgage-person-service`
- Version: `0.0.1-SNAPSHOT`
- Description: `Demo project for Spring Boot`
- Package: `com.rollingstone`

Below these fields is a 'Working sets' section with the following options:

- ☐ Add project to working sets (with a 'New...' button)
- Working sets: (dropdown menu) (with a 'Select...' button)

At the bottom of the dialog, there are four buttons: '?', '< Back', 'Next >' (highlighted with a blue border), 'Finish', and 'Cancel'.

1.3 – Do not choose anything



New Spring Starter Project

Boot Version: 1.5.1

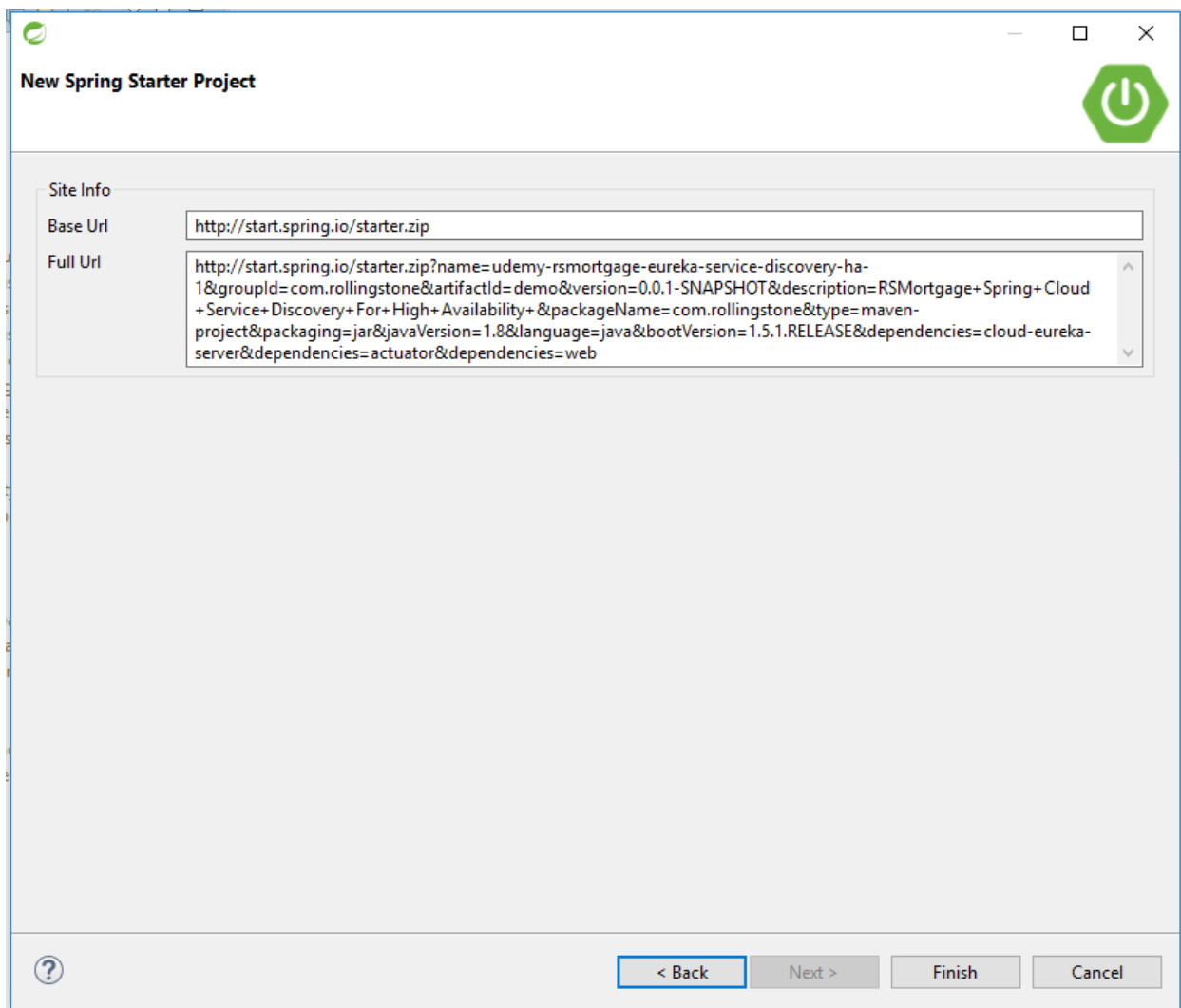
Dependencies:

Actu

- ▼ Cloud Discovery
 - ☒ Eureka Server
- ▼ Ops
 - ☒ Actuator ☐ Actuator Docs
- ▼ Web
 - ☒ Web

? < Back Next > **Finish** Cancel

1.4 – Click Finish Now




The image shows a 'New Spring Starter Project' dialog box. It has a title bar with a green icon, a minimize button, a maximize button, and a close button. The main area is titled 'New Spring Starter Project' and contains a 'Site Info' section. This section has two text fields: 'Base Url' and 'Full Url'. The 'Base Url' field contains the text 'http://start.spring.io/starter.zip'. The 'Full Url' field contains a long URL: 'http://start.spring.io/starter.zip?name=udemy-rsmortgage-eureka-service-discovery-ha-1&groupId=com.rollingstone&artifactId=demo&version=0.0.1-SNAPSHOT&description=RSMortgage+Spring+Cloud+Service+Discovery+For+High+Availability+&packageName=com.rollingstone&type=maven-project&packaging=jar&javaVersion=1.8&language=java&bootVersion=1.5.1.RELEASE&dependencies=cloud-eureka-server&dependencies=actuator&dependencies=web'. At the bottom of the dialog, there is a row of four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'. The '< Back' button is highlighted with a blue border. There is also a help icon (a question mark in a circle) on the bottom left.

New Spring Starter Project

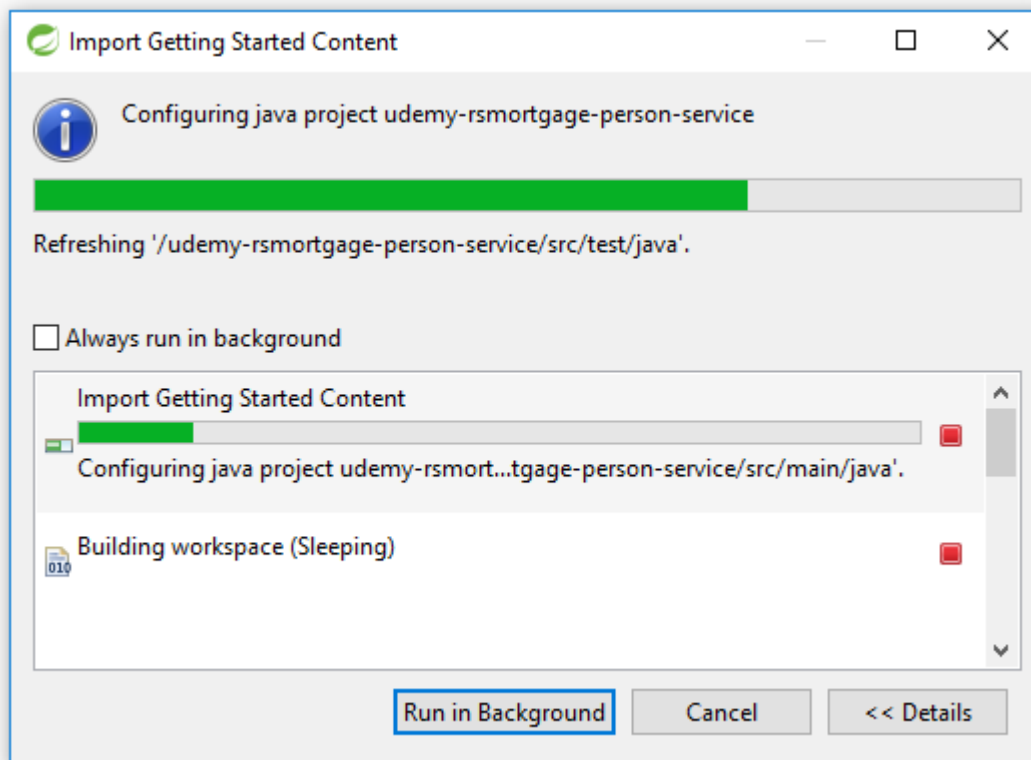
Site Info

Base Url:

Full Url:



1.5 – Let Spring Tool Suite Prepare the Project



1.6 – Verify the project identification in the pom.xml

```
<groupId>com.rollingstone</groupId>
<artifactId>udemy-rsmortgage-person-service</artifactId>
<version>0.0.1-SNAPSHOT</version>
<packaging>jar</packaging>

<name>udemy-rsmortgage-person-service</name>
<description>Demo project for Spring Boot</description>
```

1.7 – Add Parent project

```
<parent>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-parent</artifactId>
  <version>1.3.6.RELEASE</version>
</parent>
```

1.8 – Add Properties

```
<properties>
  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
  <java.version>1.8</java.version>
<start-class>com.rollingstone.Application</start-class>
  <!--      <slf4j.version>1.6.4</slf4j.version>
  <logback.version>1.0.1</logback.version> -->
</properties>
```

1.9– Add web and actuator Dependencies

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

<!-- web development, including Tomcat and spring-webmvc -->
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
</dependency>
```

1.10 – Add security and tomcat dependencies

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

<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-tomcat</artifactId>
    <scope>provided</scope>
</dependency>
```

1.11 – Add jpa orm and h2 db dependencies

```
<!-- spring-data-jpa, spring-orm and Hibernate -->
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>

<dependency>
    <groupId>com.h2database</groupId>
    <artifactId>h2</artifactId>
    <version>1.4.181</version>
</dependency>
<dependency>
    <groupId>org.hsqldb</groupId>
    <artifactId>hsqldb</artifactId>
    <scope>runtime</scope>
```

```
</dependency>
```

1.12 – Add spring test dependencies

```
<!-- spring-test, hamcrest, ... -->  
<dependency>  
  <groupId>org.springframework.boot</groupId>  
  <artifactId>spring-boot-starter-test</artifactId>  
  <scope>test</scope>  
</dependency>
```

```
<!-- attribute level json comparisons -->  
  <dependency>  
    <groupId>com.jayway.jsonpath</groupId>  
    <artifactId>json-path</artifactId>  
    <version>0.9.1</version>  
    <scope>test</scope>  
  </dependency>  
  <dependency>  
    <groupId>com.jayway.jsonpath</groupId>  
    <artifactId>json-path-assert</artifactId>  
    <version>0.9.1</version>  
    <scope>test</scope>  
  </dependency>
```

1.13 – Add Jackson databind dependencies

```
<dependency>  
  <groupId>com.fasterxml.jackson.core</groupId>  
  <artifactId>jackson-databind</artifactId>  
</dependency>
```

1.14 – Add HAL Browser dependencies

```
<dependency>  
  <groupId>org.springframework.data</groupId>  
  <artifactId>spring-data-rest-hal-browser</artifactId>  
</dependency>
```

1.15 – Add Swagger dependencies

```
<dependency>
  <groupId>io.springfox</groupId>
  <artifactId>springfox-swagger2</artifactId>
  <version>2.3.1</version>
</dependency>
<dependency>
  <groupId>io.springfox</groupId>
  <artifactId>springfox-swagger-ui</artifactId>
  <version>2.3.1</version>
</dependency>
```

1.16 – Add MySQL Driver dependencies

```
<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
  <version>5.1.40</version>
</dependency>
```

1.17 – Add maven build section

```
<build>
  <resources>
    <resource>
      <directory>src/main/resources</directory>
      <filtering>true</filtering>
    </resource>
  </resources>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-compiler-plugin</artifactId>
      <version>3.1</version>
      <configuration>
        <source>1.8</source>
        <target>1.8</target>
      </configuration>
    </plugin>
    <!-- Spring boot support -->
    <plugin>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-maven-plugin</artifactId>
      <configuration>
        <addResources>>false</addResources>
      </configuration>
    </plugin>
  </plugins>
```

```
</plugins>
</build>
```

1.18 –Add properties for application.properties resource directory

```
logging.level.org.springframework.web=INFO
logging.level.org.hibernate=ERROR
logging.file=logs/udemy-rsmortgage-person-service.log
```

1.19 –Create application.yml under resource directory

```
spring.jmx:
  enabled: false
```

```
spring.datasource:
  driverClassName: org.h2.Driver
  url: jdbc:h2:mem:bootexample;MODE=MySQL
```

```
server:
  port: 8090
```

#todo: make sure to always enable security in production

```
security:
  basic:
    enabled: false
```

#management endpoints on a separate port

```
management:
  port: 8091
  security:
    enabled: false # management port is internal only. no need to secure it.
```

#default project info followed by actual injected pom-specified values.

```
project:
  name: udemy-rsmortgage-person-service
  version: 0.1
  description: udemy-rsmortgage-person-service
info:
  build:
    artifact: ${project.artifactId}
    name: ${project.name}
    description: ${project.description}
    version: ${project.version}
```

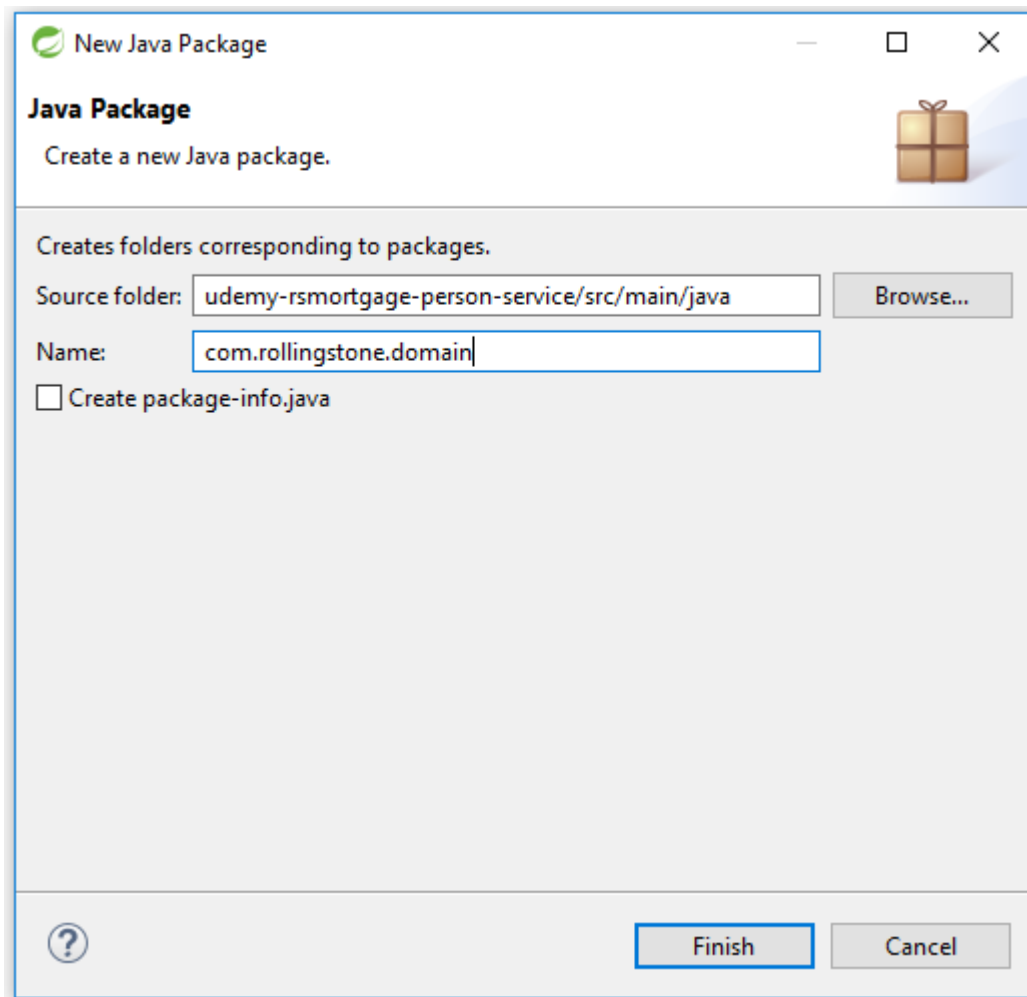
```
spring.jpa:
  hibernate.ddl-auto: create-drop

---
spring:
  profiles: mysql

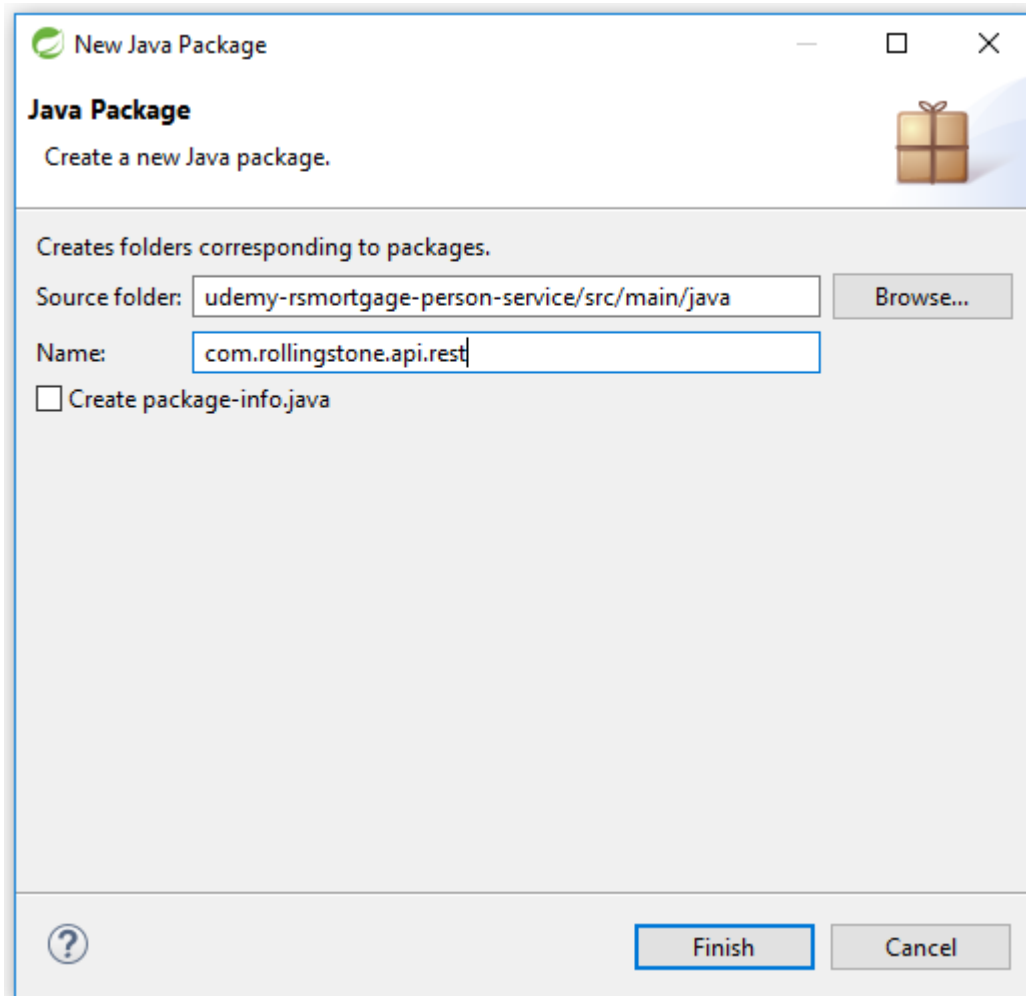
datasource:
  driverClassName: com.mysql.jdbc.Driver
  url: jdbc:mysql://localhost/rsmortgage
  username: root
  password: root

jpa:
  hibernate:
    dialect: org.hibernate.dialect.MySQLInnoDBDialect
    ddl-auto: update # todo: in non-dev environments, comment this out:
```

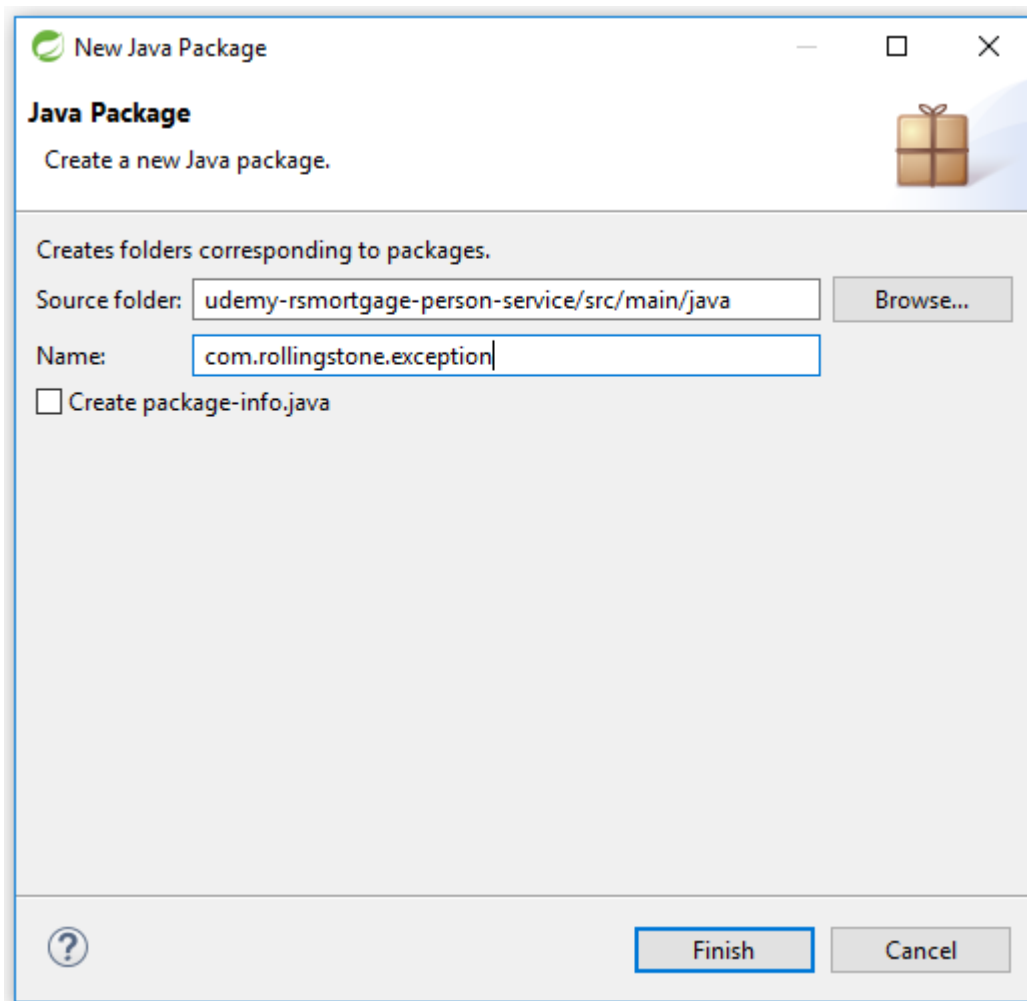

1.20 –Create domain package



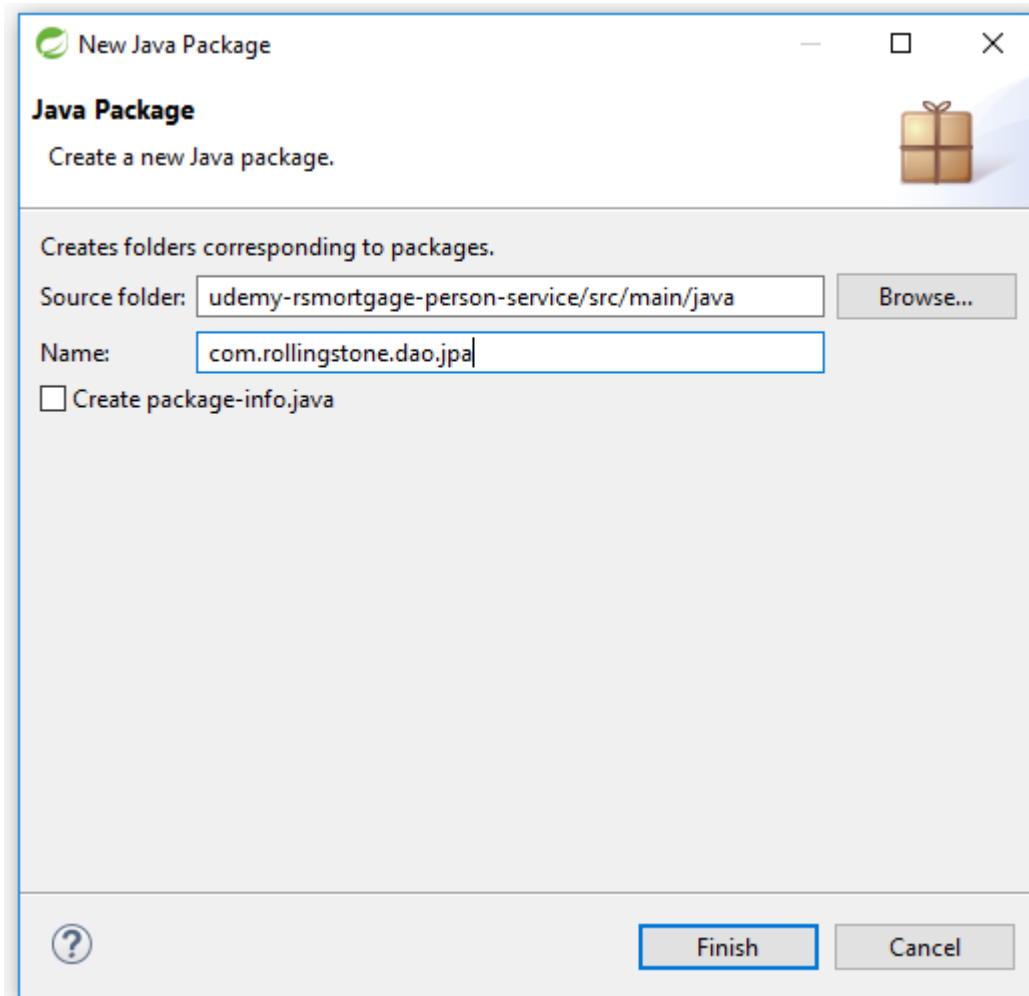
1.21 –Create api.rest package



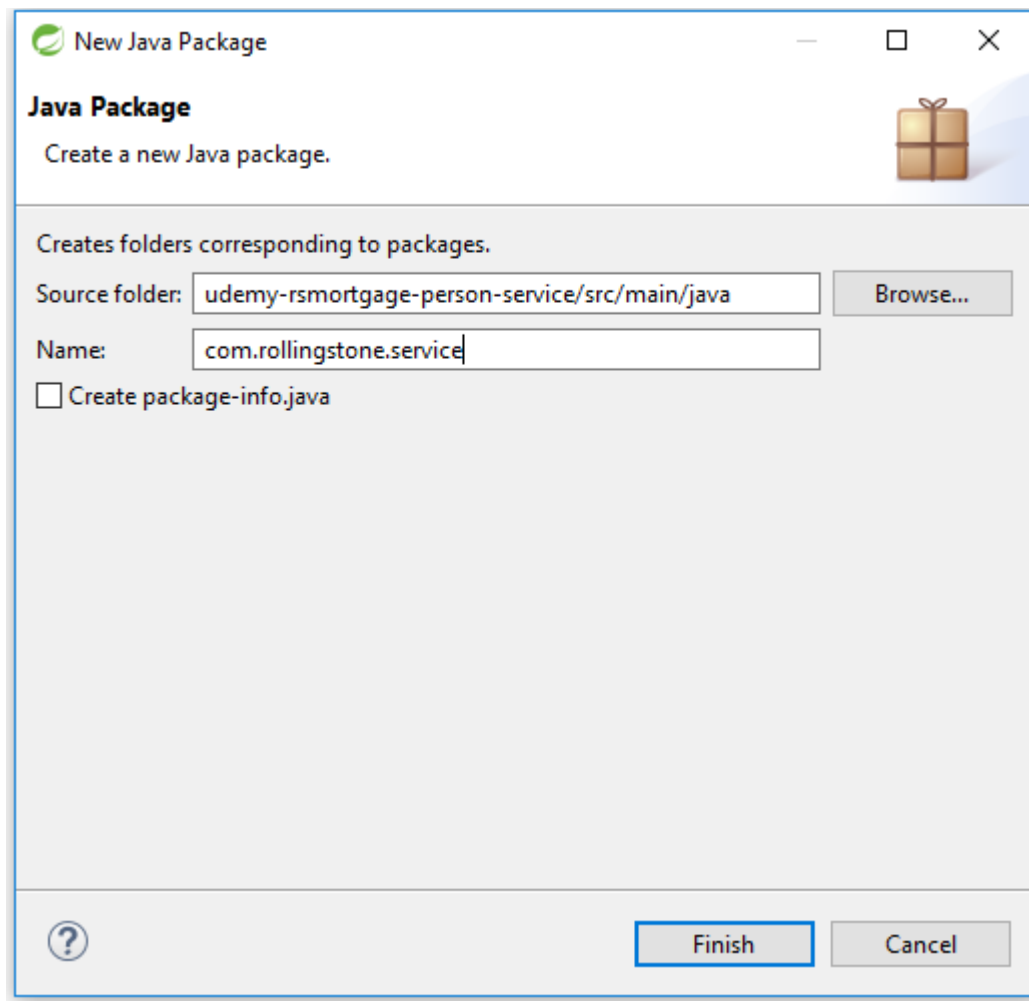
1.22 –Create exception package



1.23 –Create dao.jpa package



1.24 –Create service package



1.25 –Create Person domain class in domain package

```
package com.rollingstone.domain;

import java.util.Date;

import javax.persistence.*;
import javax.xml.bind.annotation.*;

/*
 * a simple domain entity doubling as a DTO
 */
@Entity
@Table(name = "rsmortgage_person")
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class Person {

    @Id
    @GeneratedValue
    private long id;

    @Column(nullable = false)
    private String firstName;

    @Column(nullable = false)
    private String lastName;

    @Temporal(TemporalType.DATE)
    @Column(name = "dob", unique = true, nullable = false, length = 10)
    private Date dateOfBirth;

    @Column(nullable = false)
    private int age;

    @Column(nullable = false)
    private String socialSecurityNumber;

    public Person() {
    }
}
```

1.26 –Do the following for the Person domain class

- Generate a non-default constructor using all fields
- Generate a hashCode method
- Generate an equals method
- Generate a toString method using all fields
- Generate getter and setters for all fields

1.27 –Create RESTAPIErrorInfo domain class

```
package com.rollingstone.domain;
```

```
import javax.xml.bind.annotation.XmlRootElement;
```

```
@XmlRootElement
```

```
public class RESTAPIErrorInfo {  
    public final String errorDetail;  
    public final String errorMessage;
```

```
    public RESTAPIErrorInfo(Exception ex, String detail) {  
        this.errorMessage = ex.getLocalizedMessage();  
        this.errorDetail = detail;  
    }  
}
```

1.28 –Create HTTP400Exception class in the exception pckage

```
package com.rollingstone.exception;

/**
 * for HTTP 400 Bad Request errors
 */
public final class HTTP400Exception extends RuntimeException {
    public HTTP400Exception() {
        super();
    }

    public HTTP400Exception(String message, Throwable cause) {
        super(message, cause);
    }

    public HTTP400Exception(String message) {
        super(message);
    }

    public HTTP400Exception(Throwable cause) {
        super(cause);
    }
}
```


1.29 –Create HTTP404Exception class in the exception pckage

```
package com.rollingstone.exception;

/**
 * For HTTP 404 Not Found errors
 */
public class HTTP404Exception extends RuntimeException {
    /**
     *
     */
    private static final long serialVersionUID = 1L;

    public HTTP404Exception() {
        super();
    }

    public HTTP404Exception(String message, Throwable cause) {
        super(message, cause);
    }

    public HTTP404Exception(String message) {
        super(message);
    }

    public HTTP404Exception(Throwable cause) {
        super(cause);
    }
}
```

1.30 –Create PersonRepository interface in the dao.jpa package

```
package com.rollingstone.dao.jpa;

import java.util.List;

import org.springframework.data.domain.Page;
import org.springframework.data.domain.Pageable;
import org.springframework.data.repository.PagingAndSortingRepository;

import com.rollingstone.domain.Person;

public interface PersonRepository extends PagingAndSortingRepository<Person,
Long> {
    Person findPersopnByage(int age);
    List<Person> findPersonByLastName(String lastName);
    Page findAll(Pageable pageable);
}
```

1.31 –Create ServiceProperties class in the service package

```
package com.rollingstone.service;

import org.springframework.boot.context.properties.ConfigurationProperties;
import org.springframework.stereotype.Component;
import javax.validation.constraints.NotNull;

/*
 * demonstrates how service-specific properties can be injected
 */
@ConfigurationProperties(prefix = "person.service", ignoreUnknownFields = false)
@Component
public class ServiceProperties {
    @NotNull // you can also create configurationPropertiesValidator
    private String name = "Person Service";

    @NotNull // you can also create configurationPropertiesValidator
    private String description = "Person Service Description";

    public String getName() {
        return this.name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getDescription() {
        return description;
    }

    public void setDescription(String description) {
        this.description = description;
    }
}
```

1.32 –Create PersonServiceHealth class in the service package

```
package com.rollingstone.service;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.actuate.health.Health;
import org.springframework.boot.actuate.health.HealthIndicator;
import org.springframework.stereotype.Component;

/**
 * This is an optional class used to inject application specific health check
 * into the Spring Boot health management endpoint.
 */
@Component
public class PersonServiceHealth implements HealthIndicator {

    @Autowired
    private ServiceProperties configuration;

    @Override
    public Health health() {
        return Health.up().withDetail("details",
            "{ 'internals' : 'getting close to limit', 'profile' : " +
this.configuration.getName() + " }" + this.configuration.getDescription())
            .status("itsok!").build();
    }
}
```

1.33 –Create PersonServiceEvent class in the service package

```
package com.rollingstone.service;

import org.springframework.context.ApplicationEvent;

import com.rollingstone.domain.Person;

/**
 * This is an optional class used in publishing application events.
 * This can be used to inject events into the Spring Boot audit management endpoint.
 */
public class PersonServiceEvent extends ApplicationEvent {

    Person eventPerson;
    String eventType;

    public PersonServiceEvent(Object source, String eventType, Person person) {
        super(source);
        this.eventType = eventType;
        this.eventPerson = person;
    }

    public String toString() {
        return "My PErsonService Event";
    }

    public Person getEventPerson() {
        return eventPerson;
    }

    public void setEventPerson(Person eventPerson) {
        this.eventPerson = eventPerson;
    }

    public String getEventType() {
        return eventType;
    }

    public void setEventType(String eventType) {
        this.eventType = eventType;
    }
}
```

```
}
```

1.34 –Create PersonServiceListener class in the service package

```
package com.rollingstone.service;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.context.ApplicationListener;
import org.springframework.stereotype.Component;

@Component
public class PersonEventListener implements
ApplicationListener<PersonServiceEvent>
{
    private final Logger logger = LoggerFactory.getLogger(this.getClass());

    public void onApplicationEvent(PersonServiceEvent event)
    {
        PersonServiceEvent personEvent = (PersonServiceEvent) event;

        logger.info("Person " + event.getEventType() + " with details : " +
event.getEventPerson());
    }
}
```

1.35 –Create PersonService class in the service package

```
package com.rollingstone.service;

import com.rollingstone.dao.jpa.PersonRepository;
import com.rollingstone.domain.Person;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.actuate.metrics.CounterService;
import org.springframework.boot.actuate.metrics.GaugeService;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.PageRequest;
import org.springframework.stereotype.Service;

@Service
public class PersonService {

    private static final Logger log = LoggerFactory.getLogger(PersonService.class);

    @Autowired
    private PersonRepository personRepository;

    @Autowired
    CounterService counterService;

    @Autowired
    GaugeService gaugeService;

    public PersonService() {
    }

    public Person createPerson(Person person) {
        return personRepository.save(person);
    }

    public Person getPerson(long id) {
        return personRepository.findOne(id);
    }
}
```

```

public void updatePerson(Person person) {
    personRepository.save(person);
}

public void deletePerson(Long id) {
    personRepository.delete(id);
}

public Page<Person> getAllPersons(Integer page, Integer size) {
    Page pageOfPersons = personRepository.findAll(new PageRequest(page, size));
    // example of adding to the /metrics
    if (size > 50) {
        counterService.increment("com.rollingstone.PersonService.getAll.largePayload");
    }
    return pageOfPersons;
}
}

```


1.36 –Create AbstractRestHandler class in the rest.api package

```
package com.rollingstone.api.rest;

import javax.servlet.http.HttpServletResponse;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.actuate.metrics.CounterService;
import org.springframework.context.ApplicationEventPublisher;
import org.springframework.context.ApplicationEventPublisherAware;
import org.springframework.http.HttpStatus;
import org.springframework.web.bind.annotation.ExceptionHandler;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.ResponseStatus;
import org.springframework.web.context.request.WebRequest;

import com.rollingstone.domain.RESTAPIErrorInfo;
import com.rollingstone.exception.HTTP400Exception;
import com.rollingstone.exception.HTTP404Exception;

/**
 * This class is used as a super class to be extended by all REST API "controllers".
 */
public abstract class AbstractRestHandler implements
ApplicationEventPublisherAware {

    protected final Logger log = LoggerFactory.getLogger(this.getClass());
    protected ApplicationEventPublisher eventPublisher;

    protected static final String DEFAULT_PAGE_SIZE = "100";
    protected static final String DEFAULT_PAGE_NUM = "0";

    @Autowired
    CounterService counterService;

    @ResponseStatus(HttpStatus.BAD_REQUEST)
    @ExceptionHandler(HTTP400Exception.class)
    public
    @ResponseBody
    RESTAPIErrorInfo handleDataStoreException(HTTP400Exception ex, WebRequest
request, HttpServletResponse response) {
```

```

        log.info("Converting Data Store exception to RestResponse : " + ex.getMessage());
        counterService.increment("com.rollingstone.http.400.count");
        return new RESTAPIErrorInfo(ex, "You messed up.");
    }

    @ResponseStatus(HttpStatus.NOT_FOUND)
    @ExceptionHandler(HTTP404Exception.class)
    public
    @ResponseBody
    RESTAPIErrorInfo handleResourceNotFoundException(HTTP404Exception ex,
    WebRequest request, HttpServletResponse response) {
        log.info("ResourceNotFoundException handler:" + ex.getMessage());
        counterService.increment("com.rollingstone.http.404.count");

        return new RESTAPIErrorInfo(ex, "Sorry I couldn't find it.");
    }

    @Override
    public void setApplicationEventPublisher(ApplicationEventPublisher
    applicationEventPublisher) {
        this.eventPublisher = applicationEventPublisher;
    }

    public static <T> T checkResourceFound(final T resource) {
        if (resource == null) {
            throw new HTTP404Exception("resource not found");
        }
        return resource;
    }
}

```

1.37 –Add PersonController in the rest.api package

```
package com.rollingstone.api.rest;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.http.HttpStatus;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.ResponseStatus;
import org.springframework.web.bind.annotation.RestController;

import com.rollingstone.domain.Person;
import com.rollingstone.exception.HTTP404Exception;
import com.rollingstone.service.PersonService;
import com.rollingstone.service.PersonServiceEvent;

@RestController
@RequestMapping(value = "/rsmortgage-personservice/v1/person")
public class PersonController extends AbstractRestHandler {

    @Autowired
    private PersonService personService;

    @RequestMapping(value = "",
        method = RequestMethod.POST,
        consumes = {"application/json", "application/xml"},
        produces = {"application/json", "application/xml"})
    @ResponseStatus(HttpStatus.CREATED)
    public void createPerson(@RequestBody Person person,
        HttpServletRequest request, HttpServletResponse response) {
        Person createdPerson = this.personService.createPerson(person);
        eventPublisher.publishEvent(new
        PersonServiceEvent(this,"PersonCreated",createdPerson));
        response.setHeader("Location",
        request.getRequestURL().append("/").append(createdPerson.getId()).toString());
    }
}
```

```

}

@RequestMapping(value = "",
    method = RequestMethod.GET,
    produces = {"application/json", "application/xml"})
@ResponseStatus(HttpStatus.OK)
public
@ResponseBody
Page<Person> getAllPerson(@RequestParam(value = "page", required = true,
defaultValue = DEFAULT_PAGE_NUM) Integer page,
    @RequestParam(value = "size", required = true, defaultValue =
DEFAULT_PAGE_SIZE) Integer size,
    HttpServletRequest request, HttpServletResponse response) {
    return this.personService.getAllPersons(page, size);
}

@RequestMapping(value =("/{id})",
    method = RequestMethod.GET,
    produces = {"application/json", "application/xml"})
@ResponseStatus(HttpStatus.OK)
public
@ResponseBody
Person getPerson(@PathVariable("id") Long id,
    HttpServletRequest request, HttpServletResponse response) throws
Exception {
    Person person = this.personService.getPerson(id);
    checkResourceFound(person);
    return person;
}

@RequestMapping(value =("/{id})",
    method = RequestMethod.PUT,
    consumes = {"application/json", "application/xml"},
    produces = {"application/json", "application/xml"})
@ResponseStatus(HttpStatus.NO_CONTENT)
public void updatePerson(@PathVariable("id") Long id, @RequestBody Person
person,
    HttpServletRequest request, HttpServletResponse response) {
    checkResourceFound(this.personService.getPerson(id));
    if (id != person.getId()) throw new HTTP404Exception("ID doesn't match!");
    this.personService.updatePerson(person);
}

@RequestMapping(value =("/{id})",
    method = RequestMethod.DELETE,

```

```
        produces = {"application/json", "application/xml"})
    @ResponseStatus(HttpStatus.NO_CONTENT)
    public void deletePerson(@PathVariable("id") Long id, HttpServletRequest request,
                             HttpServletResponse response) {
        checkResourceFound(this.personService.getPerson(id));
        this.personService.deletePerson(id);
    }
}
```

1.38 –Add RestControllerAspect in the com.rollingstone package

```
package com.rollingstone;

import java.util.NoSuchElementException;

import org.aspectj.lang.JoinPoint;
import org.aspectj.lang.annotation.AfterReturning;
import org.aspectj.lang.annotation.AfterThrowing;
import org.aspectj.lang.annotation.Aspect;
import org.aspectj.lang.annotation.Before;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.actuate.metrics.CounterService;
import org.springframework.stereotype.Component;

@Aspect
@Component
public class RestControllerAspect {

    private final Logger logger = LoggerFactory.getLogger(this.getClass());

    @Autowired
    CounterService counterService;

    @Before("execution(public * com.rollingstone.api.rest.*Controller.*(..))")
    public void logBeforeRestCall(JoinPoint pjp) throws Throwable {
        logger.info(":::::AOP Before REST call:::::" + pjp);
    }

    @AfterReturning("execution(public *
com.rollingstone.api.rest.*Controller.createPerson*(..))")
    public void afterCallingCreatePerson(JoinPoint pjp) {
        logger.info(":::::AOP @AfterReturning Create REST call:::::" + pjp);

        counterService.increment("com.rollingstone.api.rest.PersonController.createPerson");
    }

    @AfterReturning("execution(public *
com.rollingstone.api.rest.*Controller.getAllPerson*(..))")
    public void afterCallinggetAllPerson(JoinPoint pjp) {
        logger.info(":::::AOP @AfterReturning getAllPerson REST call:::::" + pjp);
    }
}
```

```
counterService.increment("com.rollingstone.api.rest.PersonController.getAllPerson");
}
```

```
    @AfterReturning("execution(public *
com.rollingstone.api.rest.*Controller.getPerson*(..))")
    public void afterCallinggetPerson(JoinPoint pjp) {
        logger.info(":::::AOP @AfterReturning getPErson REST call:::::" + pjp);

counterService.increment("com.rollingstone.api.rest.PersonController.getPerson");
}
```

```
    @AfterReturning("execution(public *
com.rollingstone.api.rest.*Controller.updatePerson*(..))")
    public void afterCallingupdatePerson(JoinPoint pjp) {
        logger.info(":::::AOP @AfterReturning update PErson REST call:::::" + pjp);

counterService.increment("com.rollingstone.api.rest.PersonController.updatePerson");
}
```

```
    @AfterThrowing(pointcut = "execution(public *
com.rollingstone.api.rest.*Controller.*(..))", throwing = "e")
    public void afterGetGreetingThrowsException(NoSuchElementException e) {
        counterService.increment("counter.errors.person.controller");
    }
}
```

1.39 –Add Application class in the com.rollingstone package

```
package com.rollingstone;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.EnableAutoConfiguration;
import org.springframework.boot.builder.SpringApplicationBuilder;
import org.springframework.boot.context.web.SpringBootServletInitializer;
import org.springframework.context.annotation.ComponentScan;
import org.springframework.data.jpa.repository.config.EnableJpaRepositories;

/*
 * This is the main Spring Boot application class. It configures Spring Boot, JPA,
 * Swagger
 */

@EnableAutoConfiguration // Sprint Boot Auto Configuration
@ComponentScan(basePackages = "com.rollingstone")
@EnableJpaRepositories("com.rollingstone.dao.jpa") // To segregate MongoDB and JPA
repositories. Otherwise not needed.
public class Application extends SpringBootServletInitializer {

    private static final Class<Application> applicationClass = Application.class;
    private static final Logger log = LoggerFactory.getLogger(applicationClass);

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

    @Override
    protected SpringApplicationBuilder configure(SpringApplicationBuilder application)
    {
        return application.sources(applicationClass);
    }
}
```


1.40 –Open Git Bash in project folder

MINGW64:/c:/AWS/RSMortgageWS/udemy-rsmortgage-person-service

```
Caree2@Caree2-PC MINGW64 /c:/AWS/RSMortgageWS/udemy-rsmortgage-person-service
$
```

1.41 –Run build

MINGW64:/c:/AWS/RSMortgageWS/udemy-rsmortgage-person-service

```
Caree2@Caree2-PC MINGW64 /c:/AWS/RSMortgageWS/udemy-rsmortgage-person-service
$ mvn package
```

1.42 –Build successful

MINGW64/c/AWS/RSMortgageWS/udemy-rsmortgage-person-service

```
Caree2@Caree2-PC MINGW64 /c/AWS/RSMortgageWS/udemy-rsmortgage-person-service
$ mvn package
[INFO] Scanning for projects...
[INFO]
[INFO] -----
[INFO] Building udemy-rsmortgage-person-service 0.3.0
[INFO] -----
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ udemy-rsmortgage-person-service ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Copying 2 resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ udemy-rsmortgage-person-service ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ udemy-rsmortgage-person-service ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory C:\AWS\RSMortgageWS\udemy-rsmortgage-person-service\src\test\resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ udemy-rsmortgage-person-service ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-surefire-plugin:2.18.1:test (default-test) @ udemy-rsmortgage-person-service ---
[INFO] No tests to run.
[INFO]
[INFO] --- maven-jar-plugin:2.5:jar (default-jar) @ udemy-rsmortgage-person-service ---
[INFO] Building jar: C:\AWS\RSMortgageWS\udemy-rsmortgage-person-service\target\udemy-rsmortgage-person-service-0.3.0.jar
[INFO]
[INFO] --- spring-boot-maven-plugin:1.3.6.RELEASE:repackage (default) @ udemy-rsmortgage-person-service ---
[INFO]
[INFO] BUILD SUCCESS
[INFO]
[INFO] -----
[INFO] Total time: 6.875s
[INFO] Finished at: Sun Feb 26 12:09:31 CST 2017
[INFO] Final Memory: 16M/220M
[INFO] -----
```

```
Caree2@Caree2-PC MINGW64 /c/AWS/RSMortgageWS/udemy-rsmortgage-person-service
$ |
```

1.43 –Now Run

java -jar -Dspring.profiles.active=mysql target/udemy-rsmortgage-person-service-0.3.0.jar

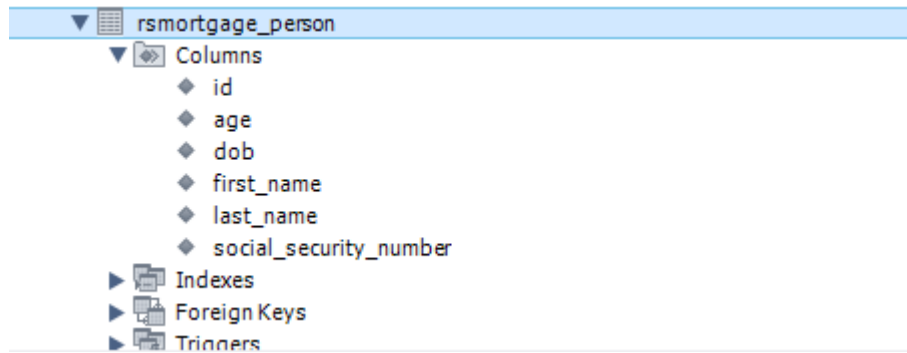
```
Caree2@Caree2-PC MINGW64 /c:/AWS/RSMortgageWS/udemy-rsmortgage-person-service
$ java -jar -Dspring.profiles.active=mysql target/udemy-rsmortgage-person-service-0.3.0.jar

:: Spring Boot ::
          (v1.3.6.RELEASE)

2017-02-26 12:11:04.308 INFO 8944 --- [main] com.rollingstone.Application : Starting Application v0.3.0 on Caree2-PC with PID 8944 (C:\AWS\RSMortgageWS\udemy-rsmortgage-person-service\target\udemy-rsmortgage-person-service-0.3.0.jar started by Caree2 in C:\AWS\RSMortgageWS\udemy-rsmortgage-person-service)
2017-02-26 12:11:04.315 INFO 8944 --- [main] com.rollingstone.Application : The following profiles are active: mysql
2017-02-26 12:11:04.433 INFO 8944 --- [main] ationConfigEmbeddedWebApplicationContext : Refreshing org.springframework.boot.context.embedded.AnnotationConfigEmbeddedWebApplicationContext@5e265ba4: startup date [Sun Feb 26 12:11:04 CST 2017]; root of context hierarchy
2017-02-26 12:11:07.604 INFO 8944 --- [main] o.s.b.f.s.DefaultListableBeanFactory : Overriding bean definition for bean 'ignoredPathsWebSecurityConfigurerAdapter' with a different definition: replacing [Root bean: class [null]; scope=; abstract=false; lazyInit=false; autowireMode=3; dependencyCheck=0; autowireCandidate=true; primary=false; factoryBeanName=org.springframework.boot.autoconfigure.security.SpringBootWebSecurityConfiguration; factoryMethodName=ignoredPathsWebSecurityConfigurerAdapter; initMethodName=null; destroyMethodName=(inferred); defined in class path resource [org/springframework/boot/autoconfigure/security/SpringBootWebSecurityConfiguration.class]] with [Root bean: class [null]; scope=; abstract=false; lazyInit=false; autowireMode=3; dependencyCheck=0; autowireCandidate=true; primary=false; factoryBeanName=org.springframework.boot.autoconfigure.ManagementWebSecurityAutoConfiguration; factoryMethodName=ignoredPathsWebSecurityConfigurerAdapter; initMethodName=null; destroyMethodName=(inferred); defined in class path resource [org/springframework/boot/autoconfigure/ManagementWebSecurityAutoConfiguration.class]]









2017-02-26 12:11:24.725 INFO 8944 --- [main] o.s.b.a.e.mvc.EndpointHandlerMapping : Mapped "[/metrics/{name:.*}]" methods=[GET], produces=[application/json] onto public java.lang.Object org.springframework.boot.actuate.endpoint.mvc.MetricsMvcEndpoint.value(java.lang.String)
2017-02-26 12:11:24.725 INFO 8944 --- [main] o.s.b.a.e.mvc.EndpointHandlerMapping : Mapped "[/metrics || /metrics.json]" methods=[GET], produces=[application/json] onto public java.lang.Object org.springframework.boot.actuate.endpoint.mvc.MetricsMvcEndpoint.invoke()
2017-02-26 12:11:24.728 INFO 8944 --- [main] o.s.b.a.e.mvc.EndpointHandlerMapping : Mapped "[/autoconfig || /autoconfig.json]" methods=[GET], produces=[application/json] onto public java.lang.Object org.springframework.boot.actuate.endpoint.mvc.EndpointMvcAdapter.invoke()
2017-02-26 12:11:24.730 INFO 8944 --- [main] o.s.b.a.e.mvc.EndpointHandlerMapping : Mapped "[/trace || /trace.json]" methods=[GET], produces=[application/json] onto public java.lang.Object org.springframework.boot.actuate.endpoint.mvc.EndpointMvcAdapter.invoke()
2017-02-26 12:11:24.731 INFO 8944 --- [main] o.s.b.a.e.mvc.EndpointHandlerMapping : Mapped "[/configprops || /configprops.json]" methods=[GET], produces=[application/json] onto public java.lang.Object org.springframework.boot.actuate.endpoint.mvc.EndpointMvcAdapter.invoke()
2017-02-26 12:11:24.733 INFO 8944 --- [main] o.s.b.a.e.mvc.EndpointHandlerMapping : Mapped "[/logfile || /logfile.json]" methods=[GET || HEAD], produces=[application/json] onto public java.lang.Object org.springframework.boot.actuate.endpoint.mvc.LogFileMvcEndpoint.invoke(javax.servlet.http.HttpServletRequest, javax.servlet.http.HttpServletResponse) throws javax.servlet.ServletException, java.io.IOException
2017-02-26 12:11:24.735 INFO 8944 --- [main] o.s.b.a.e.mvc.EndpointHandlerMapping : Mapped "[/env/{name:.*}]" methods=[GET], produces=[application/json] onto public java.lang.Object org.springframework.boot.actuate.endpoint.mvc.EnvironmentMvcEndpoint.value(java.lang.String)
2017-02-26 12:11:24.737 INFO 8944 --- [main] o.s.b.a.e.mvc.EndpointHandlerMapping : Mapped "[/env || /env.json]" methods=[GET], produces=[application/json] onto public java.lang.Object org.springframework.boot.actuate.endpoint.mvc.EndpointMvcAdapter.invoke()
2017-02-26 12:11:24.738 INFO 8944 --- [main] o.s.b.a.e.mvc.EndpointHandlerMapping : Mapped "[/info || /info.json]" methods=[GET], produces=[application/json] onto public java.lang.Object org.springframework.boot.actuate.endpoint.mvc.EndpointMvcAdapter.invoke()
2017-02-26 12:11:24.739 INFO 8944 --- [main] o.s.b.a.e.mvc.EndpointHandlerMapping : Mapped "[/health || /health.json]" produces=[application/json] onto public java.lang.Object org.springframework.boot.actuate.endpoint.mvc.HealthMvcEndpoint.invoke(java.security.Principal)
2017-02-26 12:11:24.740 INFO 8944 --- [main] o.s.b.a.e.mvc.EndpointHandlerMapping : Mapped "[/dump || /dump.json]" methods=[GET], produces=[application/json] onto public java.lang.Object org.springframework.boot.actuate.endpoint.mvc.EndpointMvcAdapter.invoke()
2017-02-26 12:11:24.741 INFO 8944 --- [main] o.s.b.a.e.mvc.EndpointHandlerMapping : Mapped "[/mappings || /mappings.json]" methods=[GET], produces=[application/json] onto public java.lang.Object org.springframework.boot.actuate.endpoint.mvc.EndpointMvcAdapter.invoke()
2017-02-26 12:11:24.751 INFO 8944 --- [main] o.s.b.a.e.mvc.EndpointHandlerMapping : Mapped "[/actuator || /actuator.json]" produces=[text/html] onto public java.lang.String org.springframework.boot.actuate.endpoint.mvc.HalBrowserMvcEndpoint.browse(javax.servlet.http.HttpServletRequest)
2017-02-26 12:11:24.752 INFO 8944 --- [main] o.s.b.a.e.mvc.EndpointHandlerMapping : Mapped "[/actuator || /actuator.json]" produces=[application/json] onto public org.springframework.hateoas.ResourceSupport org.springframework.boot.actuate.endpoint.mvc.HalJsonMvcEndpoint.links()
2017-02-26 12:11:24.789 INFO 8944 --- [main] o.s.w.s.handler.SimpleUrlHandlerMapping : Mapped URL path [/actuator/**] onto handler of type [class org.springframework.web.servlet.resource.ResourceHttpRequestHandler]
2017-02-26 12:11:24.791 INFO 8944 --- [main] o.s.w.s.handler.SimpleUrlHandlerMapping : Mapped URL path [/webjars/**] onto handler of type [class org.springframework.web.servlet.resource.ResourceHttpRequestHandler]
2017-02-26 12:11:24.791 INFO 8944 --- [main] o.s.w.s.handler.SimpleUrlHandlerMapping : Mapped URL path [/**] onto handler of type [class org.springframework.web.servlet.resource.ResourceHttpRequestHandler]
2017-02-26 12:11:24.792 INFO 8944 --- [main] o.s.w.s.handler.SimpleUrlHandlerMapping : Mapped URL path [/browser/**] onto handler of type [class org.springframework.web.servlet.resource.ResourceHttpRequestHandler]
2017-02-26 12:11:24.816 INFO 8944 --- [main] s.b.c.e.t.TomcatEmbeddedServletContainer : Mapped "[/error]" onto public java.util.Map<java.lang.String, java.lang.Object> org.springframework.boot.actuate.endpoint.mvc.ManagementErrorEndpoint.invoke()
2017-02-26 12:11:25.036 INFO 8944 --- [main] s.b.c.e.t.TomcatEmbeddedServletContainer : Tomcat started on port(s): 8091 (http)
2017-02-26 12:11:25.084 INFO 8944 --- [main] s.b.c.e.t.TomcatEmbeddedServletContainer : Tomcat started on port(s): 8090 (http)
2017-02-26 12:11:25.097 INFO 8944 --- [main] com.rollingstone.Application : Started Application in 21.538 seconds (JVM running for 22.353s)
```

1.44 –Verify Database








1.45 –Verify Database Data

user rsmortgage_person rsmortgage_person x





Don't Limit






1 • `SELECT * FROM rsmortgage.rsmortgage_person;`



<


Result Grid



Filter Rows:

Edit: 

Export/Import: 

Wrap Cell Content: 

	id	age	dob	first_name	last_name	social_security_number
▶	7	23	1993-11-03	Jacquiline	Smith	555555555
	12	23	1993-07-03	Stephen	Fleming	110112222
*	NULL	NULL	NULL	NULL	NULL	NULL

1.46 –Verify Data Retrieval API is working

<http://localhost:8090/rsmortgage-personservice/v1/person>

> <http://localhost:8090/rsmortgage-personservice/v1/person>

☒ GET ☐ POST ☐ PUT ☐ DELETE ☐ PATCH Other methods

Raw headers Headers form Headers sets Variables

Accept: application/json
Content-Type: application/json

✓ 55 bytes

200 OK 1704.00 ms

SEND

DETAILS

Raw JSON

```
{
  "content": [Array(1)
    - 0: {
      "id": 7,
      "firstName": "Jacquiline",
      "lastName": "Smith",
      "dateOfBirth": "1993-11-03",
      "age": 23,
      "socialSecurityNumber": "555555555"
    }
  ],
  "last": true,
  "totalPages": 1,
  "totalElements": 1,
  "size": 100,
  "number": 0,
  "numberOfElements": 1,
  "sort": null,
  "first": true
}
```

1.47 –Verify Data AOP for @Before and @AfterReturning

```
2017-02-26 12:16:40.854 INFO 8944 --- [nio-8090-exec-1]
com.rollingstone.RestControllerAspect : ::::AOP Before REST call::::execution(Page
com.rollingstone.api.rest.PersonController.getAllPerson(Integer,Integer,HttpServletRequest,
HttpServletResponse))
```

```
2017-02-26 12:16:41.412 INFO 8944 --- [nio-8090-exec-1]
com.rollingstone.RestControllerAspect : ::::AOP @AfterReturning getAllPerson
REST call::::execution(Page
com.rollingstone.api.rest.PersonController.getAllPerson(Integer,Integer,HttpServletRequest,
HttpServletResponse))
```

```
2017-02-26 12:16:40.748 INFO 8944 --- [nio-8090-exec-1] o.s.web.servlet.DispatcherServlet : FrameworkServlet 'dispatcherServlet': initialization completed in 61 ms
2017-02-26 12:16:40.854 INFO 8944 --- [nio-8090-exec-1] com.rollingstone.RestControllerAspect : ::::AOP Before REST call::::execution(Page com.rollingstone.api.rest.PersonController.getAllPerson(Integer,Integer,HttpServletRequest,HttpServletResponse))
2017-02-26 12:16:41.412 INFO 8944 --- [nio-8090-exec-1] com.rollingstone.RestControllerAspect : ::::AOP @AfterReturning getAllPerson REST call::::execution(Page com.rollingstone.api.rest.PersonController.getAllPerson(Integer,Integer,HttpServletRequest,HttpServletResponse))
```

1.47 –Verify Single Data Retrieval API is working

http://localhost:8090/rsmortgage-personservice/v1/person/7

GET POST PUT DELETE PATCH Other methods

Raw headers Headers form Headers sets Variables

Accept: application/json
Content-Type: application/json

55 bytes

200 OK 55.00 ms

DETAILS

Raw JSON

```
{
  "id": 7,
  "firstName": "Jacqueline",
  "lastName": "Smith",
  "dateOfBirth": "1993-11-03",
  "age": 23,
  "socialSecurityNumber": "555555555"
}
```

```
(Person com.rollingstone.api.rest.PersonController.getPerson(Long,HttpServletRequest,HttpServletResponse))
2017-02-26 12:22:03.431 INFO 8944 --- [nio-8090-exec-9] com.rollingstone.RestControllerAspect : :::::AOP Before REST call:::::execution(Person com.rolling
stone.api.rest.PersonController.getPerson(Long,HttpServletRequest,HttpServletResponse))
2017-02-26 12:22:03.437 INFO 8944 --- [nio-8090-exec-9] com.rollingstone.RestControllerAspect : :::::AOP @AfterReturning getPerson REST call:::::execution
(Person com.rollingstone.api.rest.PersonController.getPerson(Long,HttpServletRequest,HttpServletResponse))
```

```
2017-02-26 12:22:03.431 INFO 8944 --- [nio-8090-exec-9]
com.rollingstone.RestControllerAspect : :::::AOP Before REST
call:::::execution(Person
com.rollingstone.api.rest.PersonController.getPerson(Long,HttpServ
letRequest,HttpServletResponse))
2017-02-26 12:22:03.437 INFO 8944 --- [nio-8090-exec-9]
com.rollingstone.RestControllerAspect : :::::AOP
@AfterReturning getPerson REST call:::::execution(Person
com.rollingstone.api.rest.PersonController.getPerson(Long,HttpServ
letRequest,HttpServletResponse))
```


1.48 –Verify Not Found is working

```
2017-02-26 12:19:33.435 INFO 8944 --- [nio-8090-exec-3]
com.rollingstone.RestControllerAspect : :::::AOP Before REST
call:::::execution(Person
com.rollingstone.api.rest.PersonController.getPerson(Long,HttpServ
letRequest,HttpServletResponse))
2017-02-26 12:19:33.517 INFO 8944 --- [nio-8090-exec-3]
c.r.api.rest.PersonController :
ResourceNotFoundException handler:resource not found
```

> http://localhost:8090/rsmortgage-personservice/v1/person/4

☒ GET ☐ POST ☐ PUT ☐ DELETE ☐ PATCH Other methods

Raw headers Headers form Headers sets Variables

Accept: application/json
Content-Type: application/json

A✓ 55 bytes

SEND

404 Not Found 161.00 ms DETAILS

Raw JSON

```
{
  "errorDetail": "Sorry I couldn't find it.",
  "errorMessage": "resource not found"
}
```

1.49 –Verify Person Creation is working

<http://localhost:8090/rsmortgage-personservice/v1/person>

```
{
  "firstName": "Stephen",
  "lastName": "Fleming",
  "dateOfBirth": "1993-07-04",
  "age": 23,
  "socialSecurityNumber": "110112222"
}
```

Raw headers

Accept: application/json
Content-Type: application/json

Raw payload

```
{
  "firstName": "Stephen",
  "lastName": "Fleming",
  "dateOfBirth": "1993-07-04",
  "age": 23,
  "socialSecurityNumber": "110112222"
}
```

201 Created 242.00 ms

```
2017-02-26 12:24:30.242 INFO 8944 --- [nio-8090-exec-1] c.r.service.PersonEventListener : Person PersonCreated with details : Person [id=11, firstName=Stephen, lastName=Fleming, dateOfBirth=Sat Jul 03 19:00:00 CDT 1993, age=23, socialSecurityNumber=111111111]
2017-02-26 12:24:30.243 INFO 8944 --- [nio-8090-exec-1] com.rollingstone.RestControllerAspect : ::::AOP @AfterReturning Create REST call:::::execution(void com.rollingstone.api.rest.PersonController.createPerson(Person,HttpServletRequest,HttpServletResponse))
```

```
2017-02-26 12:24:30.242 INFO 8944 --- [nio-8090-exec-1]
c.r.service.PersonEventListener : Person PersonCreated
with details : Person [id=11, firstName=Stephen, lastName=Fleming,
dateOfBirth=Sat Jul 03 19:00:00 CDT 1993, age=23,
socialSecurityNumber=111111111]
Person PersonCreated with details : Person [id=11,
firstName=Stephen, lastName=Fleming, dateOfBirth=Sat Jul 03
19:00:00 CDT 1993, age=23, socialSecurityNumber=111111111]
2017-02-26 12:24:30.243 INFO 8944 --- [nio-8090-exec-1]
com.rollingstone.RestControllerAspect : ::::AOP
@AfterReturning Create REST call:::::execution(void
com.rollingstone.api.rest.PersonController.createPerson(Person,HttpServletRequest,H
ttpServletResponse))
```

The screenshot shows the SQL editor interface. The top toolbar includes icons for file operations, execution, and navigation. The SQL statement entered is:

```
1 • SELECT * FROM rsmortgage.rsmortgage_person;
```

Below the editor, the "Result Grid" tab is active. It displays the results of the query as a table with 7 columns: id, age, dob, first_name, last_name, and social_security_number. The results are as follows:

	id	age	dob	first_name	last_name	social_security_number
▶	7	23	1993-11-03	Jacquiline	Smith	55555555
	12	23	1993-07-03	Stephen	Fleming	110112222
*	NULL	NULL	NULL	NULL	NULL	NULL

The bottom toolbar contains options for filtering rows, editing, exporting/importing, and wrapping cell content.

1.49 –Verify Person Updation is working

<http://localhost:8090/rsmortgage-personservice/v1/person/11>

```
{
  "id": 11,
  "firstName": "Stephen",
  "lastName": "Fleming",
  "dateOfBirth": "1993-07-04",
  "age": 23,
  "socialSecurityNumber": "110112222"
}
```

Before

The screenshot shows a REST client interface with the following details:

- URL:** `http://localhost:8090/rsmortgage-personservice/v1/person/12`
- Method:** PUT (selected)
- Headers:** `Accept: application/json`, `Content-Type: application/json`
- Raw payload:**

```
{
  "id": 12,
  "firstName": "Stephen",
  "lastName": "Fleming",
  "dateOfBirth": "1993-07-04",
  "age": 23,
  "socialSecurityNumber": "333333333"
}
```
- Response:** 204 No Content, 109.00 ms

ID missing in payload

> http://localhost:8090/rsmortgage-personservice/v1/person/12

GET

POST

PUT

DELETE

PATCH

Other methods

application/json

Raw headers

Headers form

Headers sets

Variables

Accept: application/json

Content-Type: application/json

A✓

55 bytes

Raw payload

Data form

Files

{
 "firstName": "Stephen",
 "lastName": "Fleming",
 "dateOfBirth": "1993-07-04",
 "age": 23,
 "socialSecurityNumber": "333333333"
}

SEND

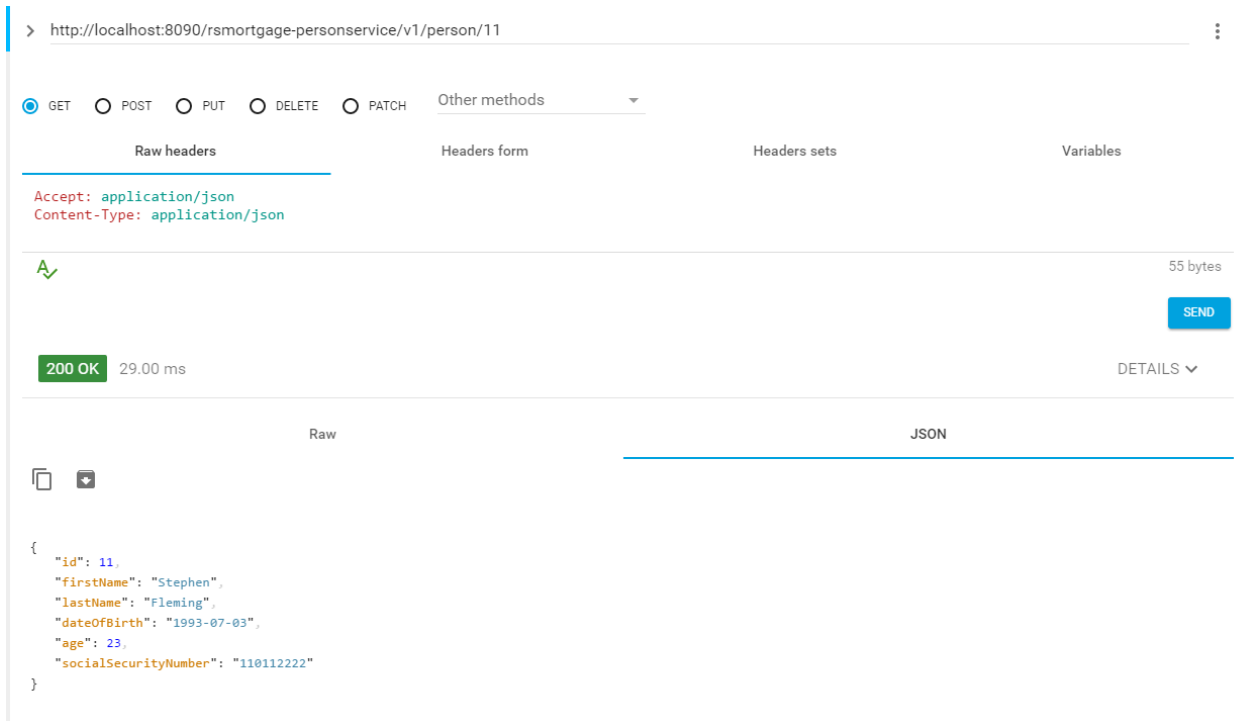
404 Not Found

34.00 ms

DETAILS

73

After



```
2017-02-26 12:30:38.625 INFO 8944 --- [nio-8090-exec-7]
com.rollingstone.RestControllerAspect : ::::AOP @AfterReturning update PErson
REST call:::::execution(void
com.rollingstone.api.rest.PersonController.updatePerson(Long,Person,HttpServletReq
uest,HttpServletResponse))
2017-02-26 12:31:07.060 INFO 8944 --- [nio-8090-exec-9]
com.rollingstone.RestControllerAspect : ::::AOP Before REST
call:::::execution(Person
com.rollingstone.api.rest.PersonController.getPerson(Long,HttpServletRequest,HttpS
ervletResponse))
2017-02-26 12:31:07.067 INFO 8944 --- [nio-8090-exec-9]
com.rollingstone.RestControllerAspect : ::::AOP @AfterReturning getPErson REST
call:::::execution(Person
com.rollingstone.api.rest.PersonController.getPerson(Long,HttpServletRequest,HttpS
ervletResponse))
```

```

one.api.rest.PersonController.updatePerson(Long, Person, HttpServletRequest, HttpServletResponse))
2017-02-26 12:27:38.231 INFO 8944 --- [nio-8090-exec-5] c.r.api.rest.PersonController : ResourceNotFoundException handler: ID doesn't match!
2017-02-26 12:30:38.374 INFO 8944 --- [nio-8090-exec-7] com.rollingstone.RestControllerAspect : :::::AOP Before REST call:::::execution(void com.rollingst
one.api.rest.PersonController.updatePerson(Long, Person, HttpServletRequest, HttpServletResponse))
2017-02-26 12:30:38.625 INFO 8944 --- [nio-8090-exec-7] com.rollingstone.RestControllerAspect : :::::AOP @AfterReturning update Person REST call:::::execu
tion(void com.rollingstone.api.rest.PersonController.updatePerson(Long, Person, HttpServletRequest, HttpServletResponse))
2017-02-26 12:31:07.060 INFO 8944 --- [nio-8090-exec-9] com.rollingstone.RestControllerAspect : :::::AOP Before REST call:::::execution(Person com.rolling
stone.api.rest.PersonController.getPerson(Long, HttpServletRequest, HttpServletResponse))
2017-02-26 12:31:07.067 INFO 8944 --- [nio-8090-exec-9] com.rollingstone.RestControllerAspect : :::::AOP @AfterReturning getPerson REST call:::::execution
(Person com.rollingstone.api.rest.PersonController.getPerson(Long, HttpServletRequest, HttpServletResponse))

```

1.50 –Verify Person Deletion is working

> http://localhost:8090/rsmortgage-personservice/v1/person/11

☐ GET
 ☐ POST
 ☐ PUT
 ☒ DELETE
 ☐ PATCH
 Other methods
 application/json

Raw headers Headers form Headers sets Variables

Accept: application/json
Content-Type: application/json

✓ 55 bytes

Raw payload Data form Files

```

{
  "id": 11,
  "firstName": "Stephen",
  "lastName": "Fleming",
  "dateOfBirth": "1993-07-04",
  "age": 23,
  "socialSecurityNumber": "110112222"
}

```

SEND

204 No Content 153.00 ms DETAILS

Not Found

> http://localhost:8090/rsmortgage-personservice/v1/person/11

☒ GET
 ☐ POST
 ☐ PUT
 ☐ DELETE
 ☐ PATCH
 Other methods

Raw headers Headers form Headers sets Variables

Accept: application/json
Content-Type: application/json

✓ 55 bytes

SEND

404 Not Found 28.00 ms DETAILS

Raw JSON

```

{
  "errorDetail": "Sorry I couldn't find it.",
  "errorMessage": "resource not found"
}

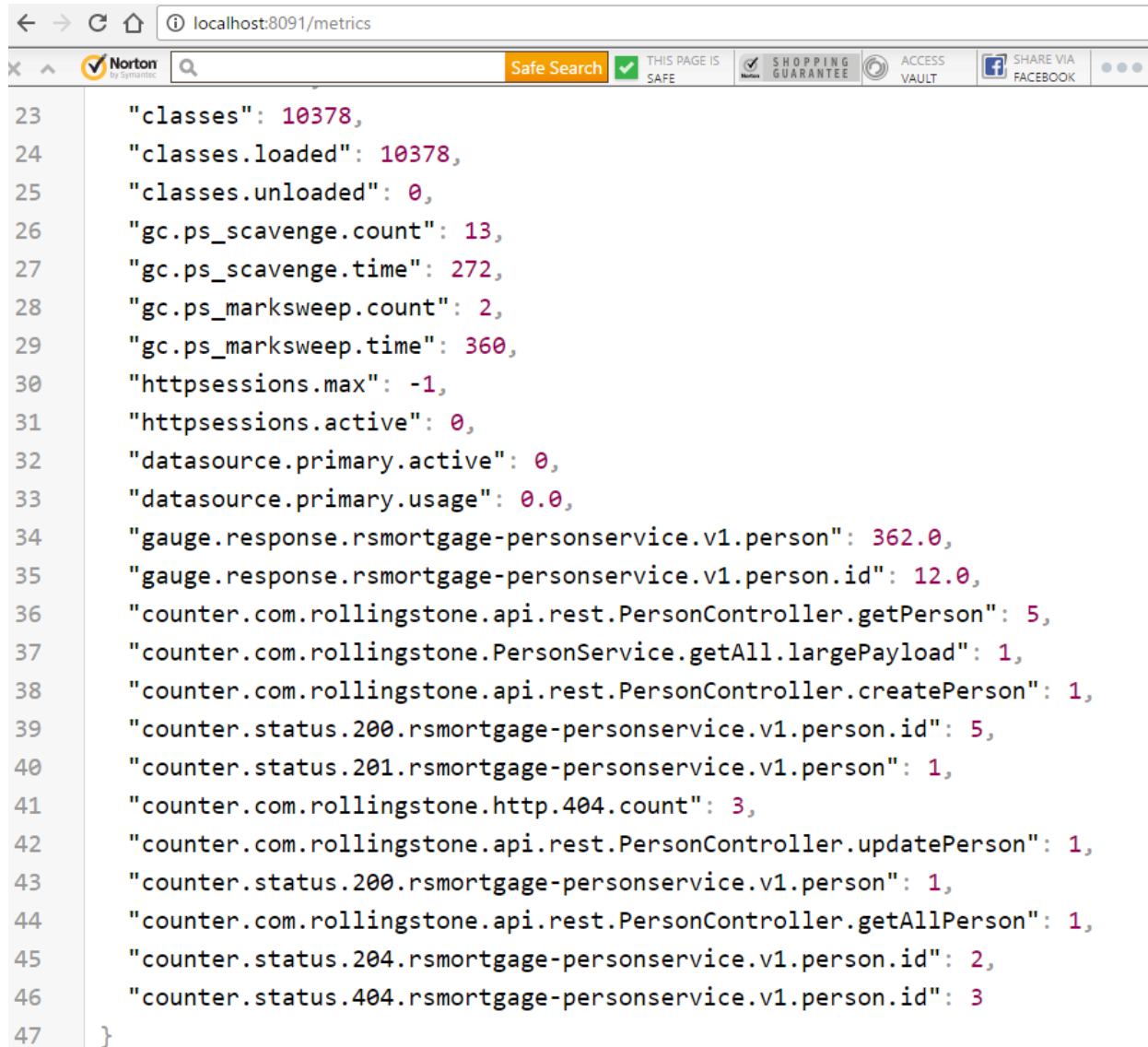
```

1.51 –Custom Health



```
1 // 20170226100155
2 // http://localhost:8091/health
3
4 {
5   "status": "UP",
6   "personServiceHealth": {
7     "status": "itsok!",
8     "details": "{ 'internals' : 'getting close to limit', 'profile' : 'Empty' }"
9   },
10  "diskSpace": {
11    "status": "UP",
12    "total": 718165323776,
13    "free": 378761715712,
14    "threshold": 10485760
15  },
16  "db": {
17    "status": "UP",
18    "database": "MySQL",
19    "hello": 1
20  }
21 }
```

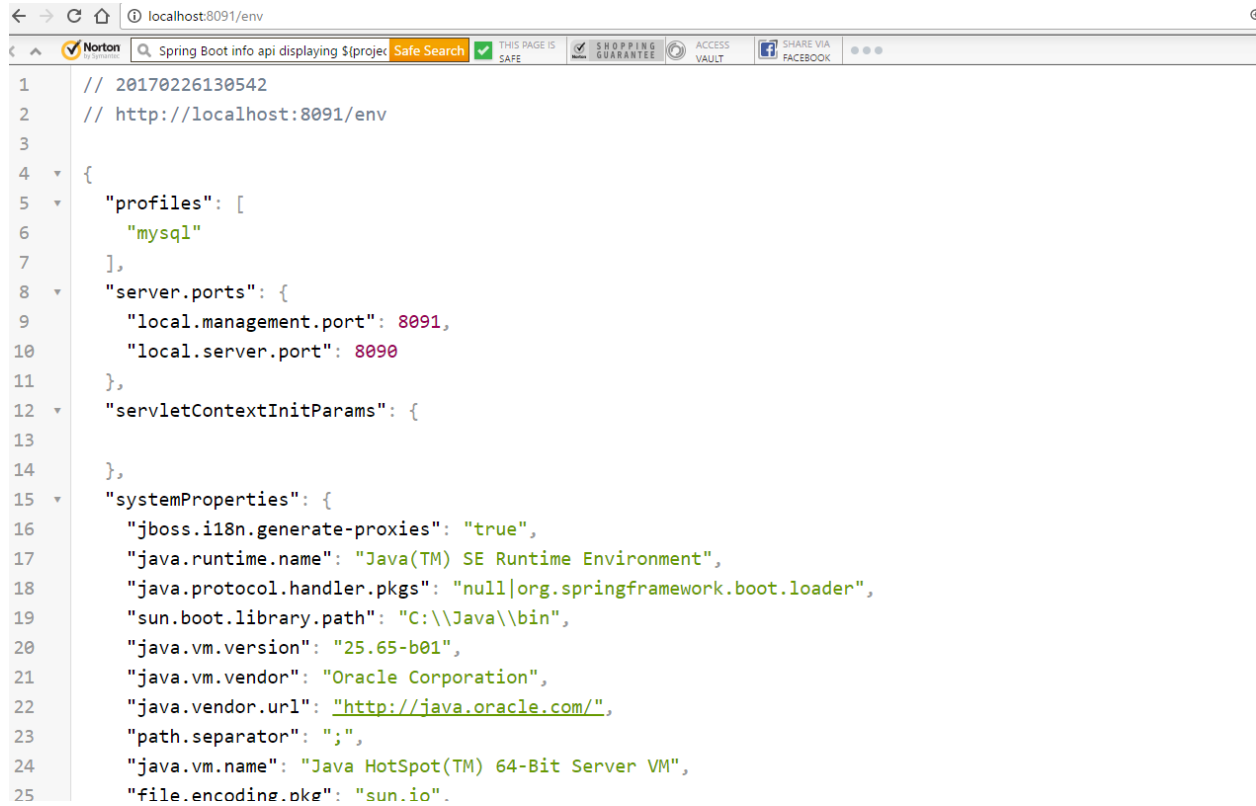

1.54 – Metrics and Custom Metrics



```
23  "classes": 10378,  
24  "classes.loaded": 10378,  
25  "classes.unloaded": 0,  
26  "gc.ps_scavenge.count": 13,  
27  "gc.ps_scavenge.time": 272,  
28  "gc.ps_marksweep.count": 2,  
29  "gc.ps_marksweep.time": 360,  
30  "httpsessions.max": -1,  
31  "httpsessions.active": 0,  
32  "datasource.primary.active": 0,  
33  "datasource.primary.usage": 0.0,  
34  "gauge.response.rsmortgage-personservice.v1.person": 362.0,  
35  "gauge.response.rsmortgage-personservice.v1.person.id": 12.0,  
36  "counter.com.rollingstone.api.rest.PersonController.getPerson": 5,  
37  "counter.com.rollingstone.PersonService.getAll.largePayload": 1,  
38  "counter.com.rollingstone.api.rest.PersonController.createPerson": 1,  
39  "counter.status.200.rsmortgage-personservice.v1.person.id": 5,  
40  "counter.status.201.rsmortgage-personservice.v1.person": 1,  
41  "counter.com.rollingstone.http.404.count": 3,  
42  "counter.com.rollingstone.api.rest.PersonController.updatePerson": 1,  
43  "counter.status.200.rsmortgage-personservice.v1.person": 1,  
44  "counter.com.rollingstone.api.rest.PersonController.getAllPerson": 1,  
45  "counter.status.204.rsmortgage-personservice.v1.person.id": 2,  
46  "counter.status.404.rsmortgage-personservice.v1.person.id": 3  
47  }
```

1.55 – /env endpoint

<http://localhost:8091/env>



```
1 // 20170226130542
2 // http://localhost:8091/env
3
4 {
5   "profiles": [
6     "mysql"
7   ],
8   "server.ports": {
9     "local.management.port": 8091,
10    "local.server.port": 8090
11  },
12  "servletContextInitParams": {
13
14  },
15  "systemProperties": {
16    "jboss.i18n.generate-proxies": "true",
17    "java.runtime.name": "Java(TM) SE Runtime Environment",
18    "java.protocol.handler.pkgs": "null|org.springframework.boot.loader",
19    "sun.boot.library.path": "C:\\Java\\bin",
20    "java.vm.version": "25.65-b01",
21    "java.vm.vendor": "Oracle Corporation",
22    "java.vendor.url": "http://java.oracle.com/",
23    "path.separator": ";",
24    "java.vm.name": "Java HotSpot(TM) 64-Bit Server VM",
25    "file.encoding.pkg": "sun.io".
```

1.56 – /beans endpoint

<http://localhost:8091/beans>



The screenshot shows a web browser window with the address bar displaying 'localhost:8091/beans'. The page content is a JSON response from a Spring Boot API. The response is a list of beans, with the first bean being an 'application' and the second being a 'personController'. The 'application' bean has a context of 'application:mysql:8090', no parent, and no dependencies. The 'personController' bean has a context of 'application:mysql:8090', no parent, and a dependency on 'counterService'.

```
1 // 20170226130716
2 // http://localhost:8091/beans
3
4 [
5   {
6     "context": "application:mysql:8090",
7     "parent": null,
8     "beans": [
9       {
10        "bean": "application",
11        "scope": "singleton",
12        "type": "com.rollingstone.Application",
13        "resource": "null",
14        "dependencies": [
15
16        ]
17      },
18      {
19        "bean": "personController",
20        "scope": "singleton",
21        "type": "com.rollingstone.api.rest.PersonController$$EnhancerBySpringCGLIB$$5542c984",
22        "resource": "URL [jar:file:/C:/AWS/RSMortgageWS/udemy-rsmortgage-person-service/target/udemy-rsmortgage-person-service-0.3.0.jar!/com/rollingstone/api/rest/PersonController.class]",
23        "dependencies": [
24          "counterService",
```

1.57 – /actuator endpoint

http://localhost:8091/actuator/#!/actuator

← ↻ 🔍

localhost:8091/actuator/#!/actuator

🔍 ☆

Norton

Spring Boot info api displaying \$projec

Safe Search

THIS PAGE IS SAFE

SHIPPING GUARANTEE

ACCESS VAULT

SHARE VIA FACEBOOK

The HAL Browser

Go To Entry Point

About The HAL Browser

Explorer

/actuator

Go!

Custom Request Headers

Properties

{ }

Links

rel	title	name / index	docs	GET	NON-GET
self				➔	!
autoconfig				➔	!
beans				➔	!
dump				➔	!

Inspector

Response Headers

200 OK

Date: Sun, 26 Feb 2017 19:09:02 GMT

Server: Apache-Coyote/1.1

























Transfer-Encoding: chunked

Content-Type: application/json

Response Body

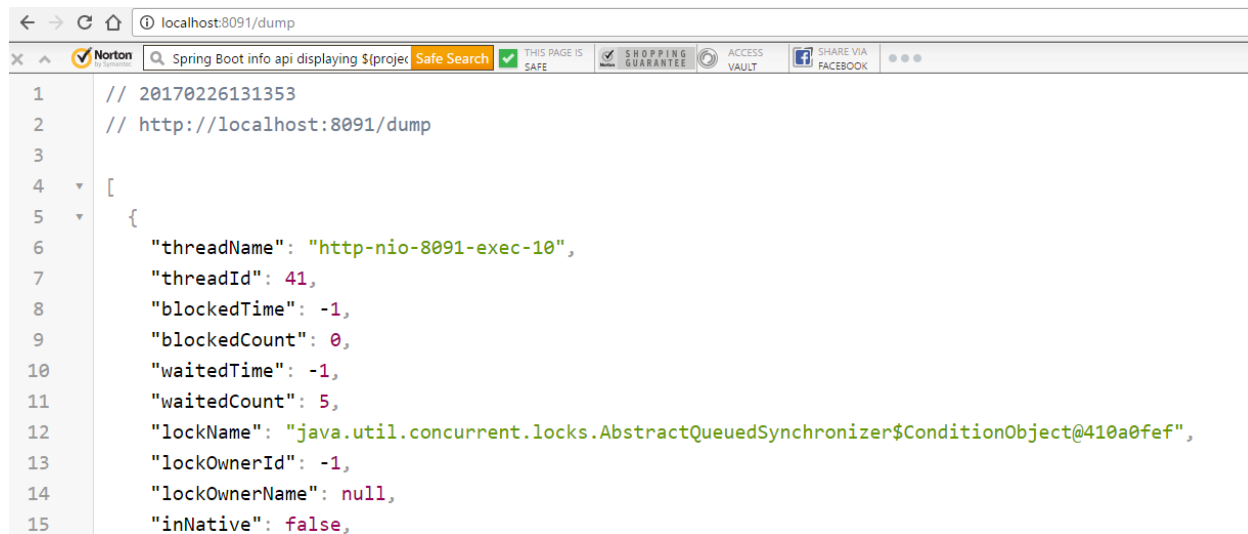
{
 "_links": {
 "self": {
 "href": "http://localhost:8091/actuator"
 },
 "autoconfig": {
 "href": "http://localhost:8091/autoconfig"
 },
 "beans": {
 "href": "http://localhost:8091/beans"
 },
 "dump": {

Links

rel	title	name / index	docs	GET	NON-GET
self					
autoconfig					
beans					
dump					
health					
mappings					
metrics					
trace					
info					
env					
logfile					
configprops					

1.58 – /dump endpoint

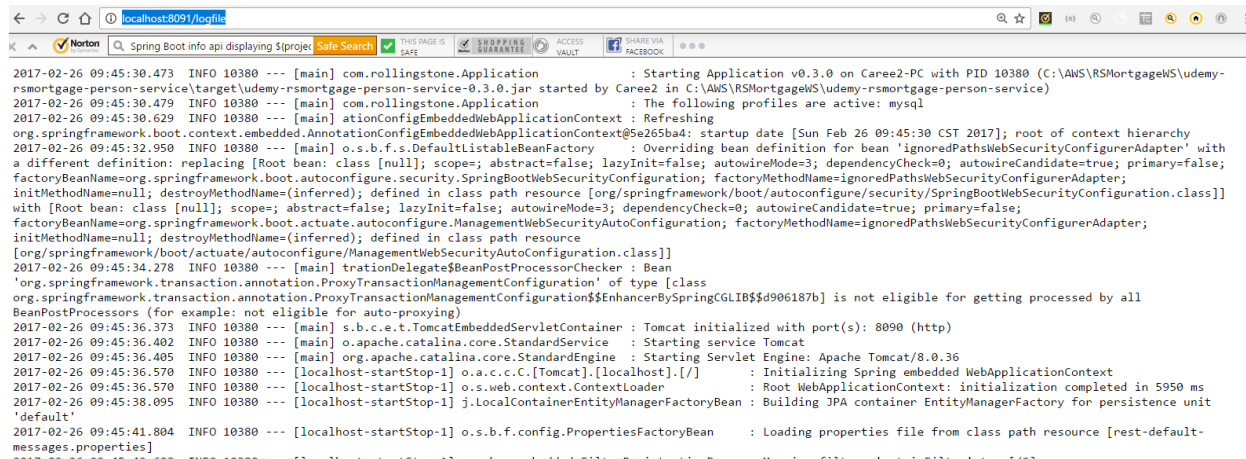
http://localhost:8091/dump



```
1 // 20170226131353
2 // http://localhost:8091/dump
3
4 [
5   {
6     "threadName": "http-nio-8091-exec-10",
7     "threadId": 41,
8     "blockedTime": -1,
9     "blockedCount": 0,
10    "waitedTime": -1,
11    "waitedCount": 5,
12    "lockName": "java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject@410a0fef",
13    "lockOwnerId": -1,
14    "lockOwnerName": null,
15    "inNative": false,
```

1.59 – /logfile endpoint

<http://localhost:8091/logfile>



```
2017-02-26 09:45:30.473 INFO 10380 --- [main] com.rollingstone.Application : Starting Application v0.3.0 on Caree2-PC with PID 10380 (C:\AWS\RSMortgageWS\udemy-rsmortgage-person-service\target\udemy-rsmortgage-person-service-0.3.0.jar started by Caree2 in C:\AWS\RSMortgageWS\udemy-rsmortgage-person-service)
2017-02-26 09:45:30.479 INFO 10380 --- [main] com.rollingstone.Application : The following profiles are active: mysql
2017-02-26 09:45:30.629 INFO 10380 --- [main] ationConfigEmbeddedWebApplicationContext : Refreshing
org.springframework.boot.context.embedded.AnnotationConfigEmbeddedWebApplicationContext@5e265ba4: startup date [Sun Feb 26 09:45:30 CST 2017]; root of context hierarchy
2017-02-26 09:45:32.950 INFO 10380 --- [main] o.s.b.f.s.DefaultListableBeanFactory : Overriding bean definition for bean 'ignoredPathsWebSecurityConfigurerAdapter' with
a different definition: replacing [Root bean: class [null]; scope=; abstract=false; lazyInit=false; autowireMode=3; dependencyCheck=0; autowireCandidate=true; primary=false;
factoryBeanName=org.springframework.boot.autoconfigure.security.SpringBootWebSecurityConfiguration; factoryMethodName=ignoredPathsWebSecurityConfigurerAdapter;
initMethodName=null; destroyMethodName=(inferred); defined in class path resource [org/springframework/boot/autoconfigure/security/SpringBootWebSecurityConfiguration.class]]
with [Root bean: class [null]; scope=; abstract=false; lazyInit=false; autowireMode=3; dependencyCheck=0; autowireCandidate=true; primary=false;
factoryBeanName=org.springframework.boot.actuate.autoconfigure.ManagementWebSecurityAutoConfiguration; factoryMethodName=ignoredPathsWebSecurityConfigurerAdapter;
initMethodName=null; destroyMethodName=(inferred); defined in class path resource
[org/springframework/boot/actuate/autoconfigure/ManagementWebSecurityAutoConfiguration.class]]
2017-02-26 09:45:34.278 INFO 10380 --- [main] trationDelegate$BeanPostProcessorChecker : Bean
'org.springframework.transaction.annotation.ProxyTransactionManagementConfiguration$$EnhancerBySpringCGLIB$$d906187b' is not eligible for getting processed by all
BeanPostProcessors (for example: not eligible for auto-proxying)
2017-02-26 09:45:36.373 INFO 10380 --- [main] s.b.c.e.t.TomcatEmbeddedServletContainer : Tomcat initialized with port(s): 8090 (http)
2017-02-26 09:45:36.402 INFO 10380 --- [main] o.apache.catalina.core.StandardService : Starting service Tomcat
2017-02-26 09:45:36.405 INFO 10380 --- [main] org.apache.catalina.core.StandardEngine : Starting Servlet Engine: Apache Tomcat/8.0.36
2017-02-26 09:45:36.570 INFO 10380 --- [localhost-startStop-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2017-02-26 09:45:36.570 INFO 10380 --- [localhost-startStop-1] o.s.web.context.ContextLoader : Root WebApplicationContext: initialization completed in 5950 ms
2017-02-26 09:45:38.095 INFO 10380 --- [localhost-startStop-1] j.LocalContainerEntityManagerFactoryBean : Building JPA container EntityManagerFactory for persistence unit
'default'
2017-02-26 09:45:41.804 INFO 10380 --- [localhost-startStop-1] o.s.b.f.config.PropertiesFactoryBean : Loading properties file from class path resource [rest-default-
messages.properties]
```

1.60 –Conclusion

This document listed the steps as well as provided the explanation of creating a Spring Boot application based on Spring Cloud Service Discovery.