

Workshop Hands On – Spring Boot to PCF

The goal of this workshop is to guide you towards building your first Spring Boot microservice application, complete with its own backing database. You will also go through the paces of deploying this app to Pivotal Application Service and experience first hand its ease and benefits. You will also learn how you can easily monitor your microservice application simply by adding a dependency.

Note: The full source code for this workshop is in GitHub. You can access it publicly at <https://github.com/derrick81/springboot-to-pcf>.

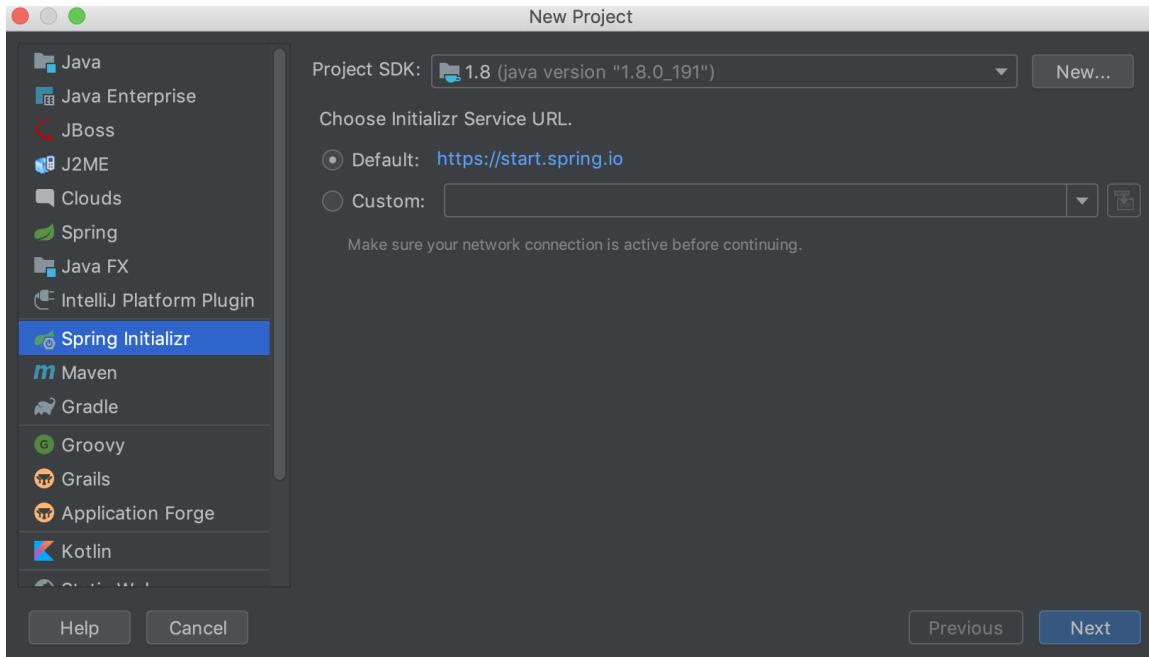
Contents

1. Your first Spring Boot microservice	2
2. Add monitoring to your microservice	9
3. Deploying and running on Pivotal Application Service	12
4. Creating and binding a service on demand	16
5. Scaling your app	21
6. Monitoring and troubleshooting your app	30

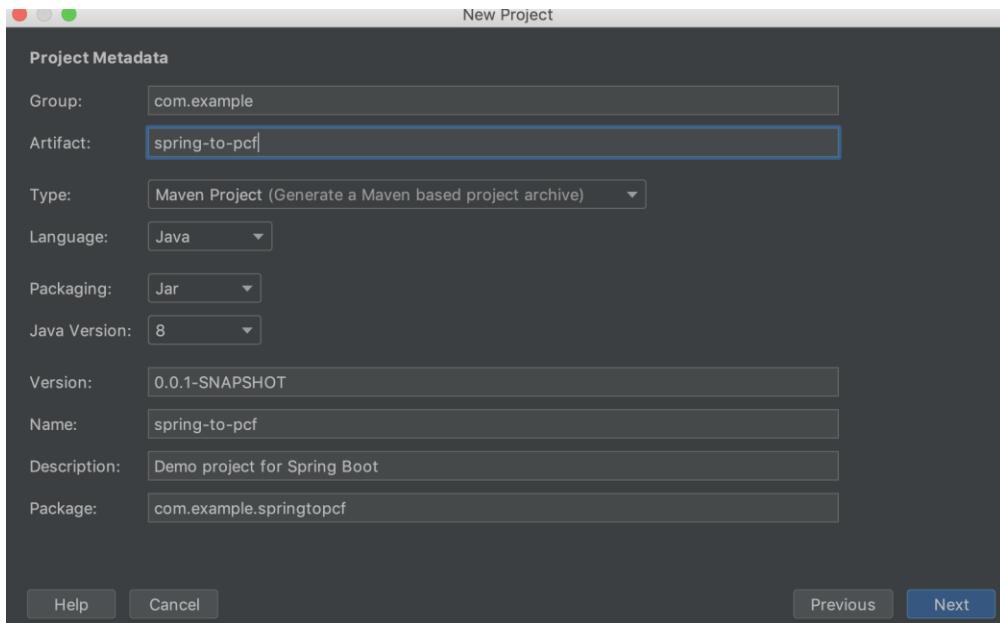
1. Your first Spring Boot microservice

- Start up the IntelliJ IDE and select the “Create New Project” option. Select “Spring Initializr” on the left hand panel and click “Next”.

Note: Alternatively you can also create the project using your browser by going to <http://start.spring.io>. However, you will have to import the project into your IDE if you do so.



- Enter “spring-to-pcf” as the artifact name, leaving the rest of the fields to their default values.

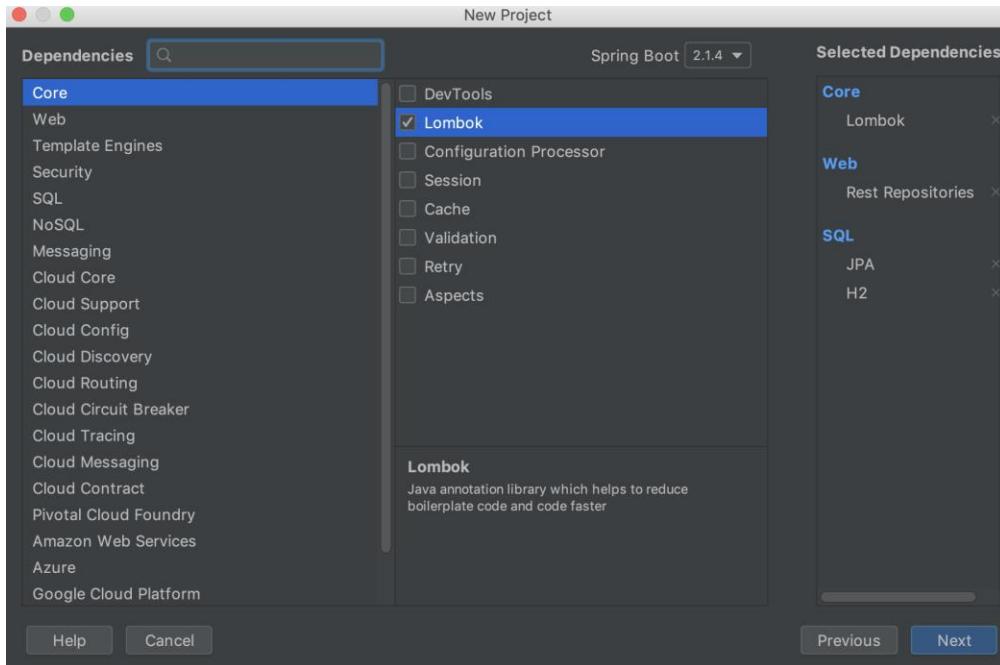




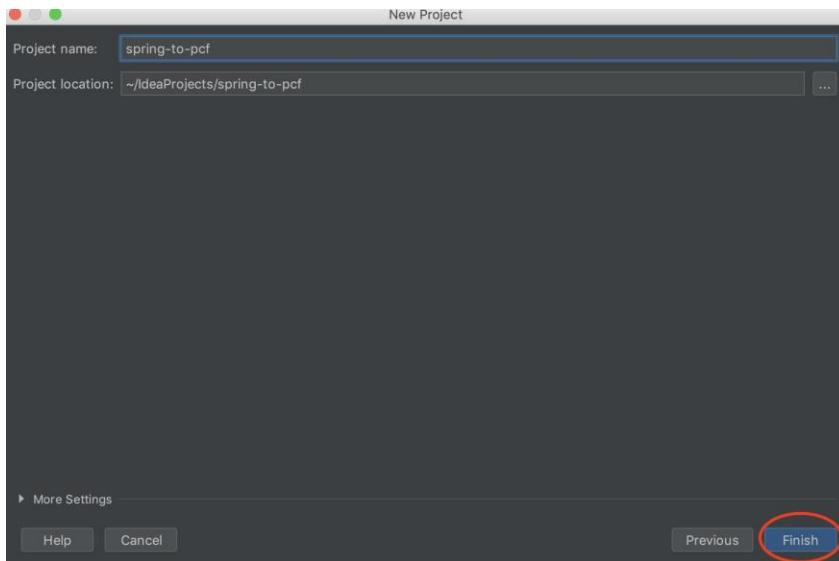
3. Select the following dependencies:

- H2
- JPA
- Rest Repositories
- Lombok

You can use the search function to quickly identify these dependencies.
Click “Next” to continue.



4. Click “Finish” and wait for Maven to complete downloading the required JAR files.





5. Open the SpringToPcfApplication file. Take note of this annotation:
 @SpringBootApplication
 You can find out more about this annotation at <https://docs.spring.io/spring-boot/docs/current/reference/html/using-boot-using-springbootapplication-annotation.html>
 In this application, Spring Boot will scan the class path and given H2 JAR files exist will auto configure a Data Source for us out of the box. Settings can be overridden using the “application.properties” or “application.yml” file.

```
1 package com.example.springtopcf;
2
3 import org.springframework.boot.SpringApplication;
4 import org.springframework.boot.autoconfigure.SpringBootApplication;
5
6 @SpringBootApplication
7 public class SpringToPcfApplication {
8
9     public static void main(String[] args) {
10         SpringApplication.run(SpringToPcfApplication.class, args);
11     }
12 }
13
14 }
```

6. Set the microservice application name by adding the following property to application.properties:

```
spring.application.name=My First Spring Boot Application
```

```
1 spring.application.name=My First Spring Boot Application
```

7. Add a JPA entity to the project by adding a new Java class named “Employee”. Here is the code for the class:

```
package com.example.springtopcf;

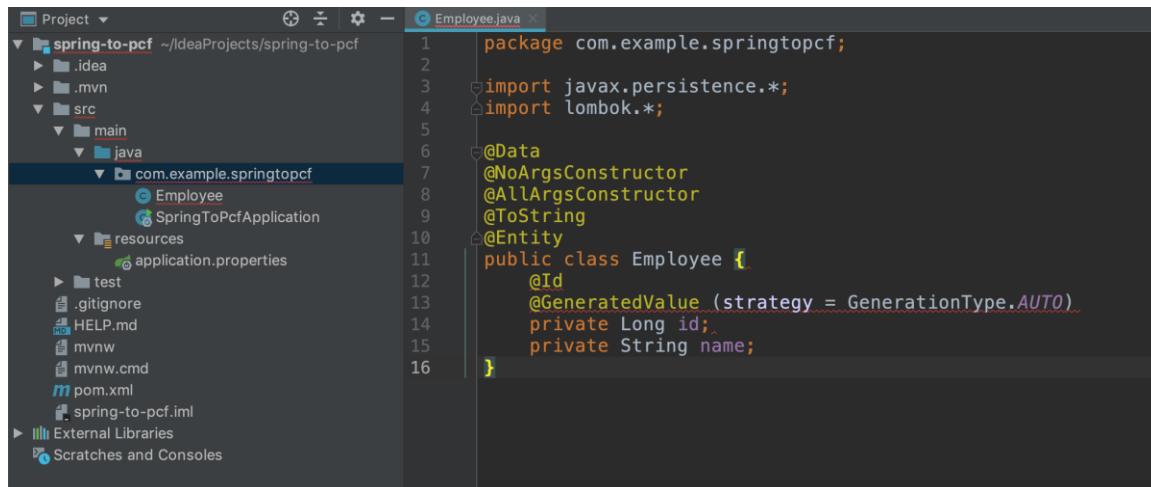
import javax.persistence.*;
import lombok.*;

@Data
@NoArgsConstructor
@AllArgsConstructor
@ToString
@Entity
public class Employee {

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)

    private Long id;

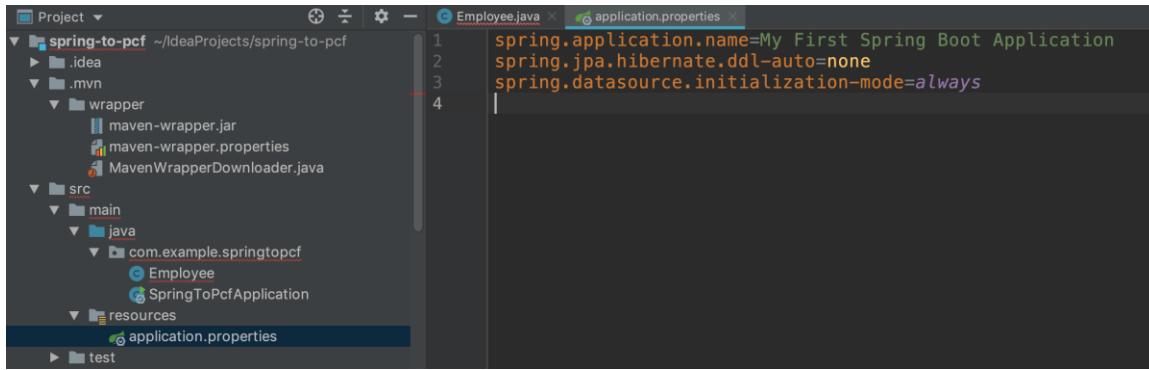
    private String name;
}
```



8. By virtue of the Spring Boot starter, Hibernate libraries have been added in as the JPA provider. Now we will specify some of the Hibernate properties. Add the following two lines to application.properties:

```
spring.jpa.hibernate.ddl-auto=none
spring.datasource.initialization-mode=always
```

The resulting application.properties should look like this:

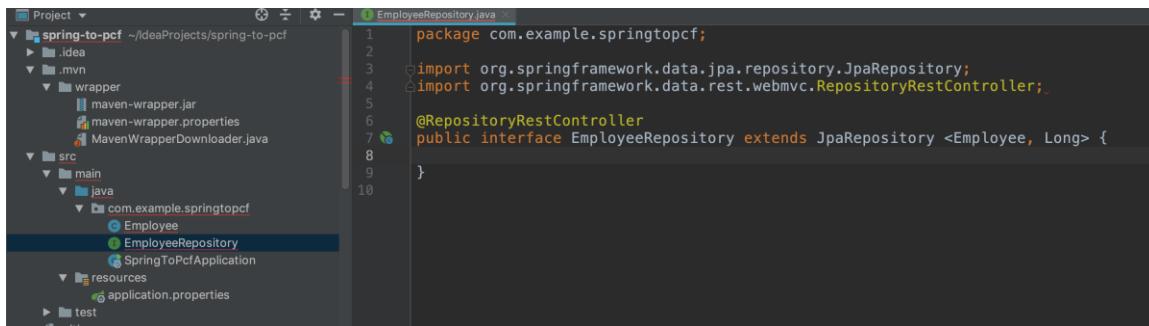


9. Next we will create REST endpoints for Employee repository. We will do so using REST Repositories. With the "@RepositoryRestResource" annotation, Spring Data REST exposes a collection resource named after the uncapitalized, pluralized version of the domain class in this case "employee". This is all done using Spring HATEOAS which it's goal is all about giving the client as much information as possible to be able to navigate the API. Here is the code for the interface:

```
package com.example.springtopcf;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.rest.webmvc.RepositoryRestController;

@RepositoryRestController
public interface EmployeeRepository extends JpaRepository <Employee, Long> {
```



10. Now we will add some scripts that will provide for both the schema and the records.

For the schema, please add schema.sql to the resources folder with the following script:

```
DROP TABLE IF EXISTS employee;

create table employee
(id bigint NOT NULL AUTO_INCREMENT PRIMARY KEY,
name VARCHAR(30))AUTO_INCREMENT=1;
```

The screenshot shows the IntelliJ IDEA interface with the project 'SpringToPcf' open. The left sidebar displays the project structure with folders for .idea, .mvn, src (containing main and java), resources (containing application.properties, data.sql, and schema.sql), and test. The right panel shows two tabs: 'schema.sql' and 'data.sql'. The 'schema.sql' tab contains the following SQL code:

```

1 DROP TABLE IF EXISTS employee;
2
3 create table employee
4   (id bigint NOT NULL AUTO_INCREMENT PRIMARY KEY,
5    name VARCHAR(30))AUTO_INCREMENT=1;
6

```

For the records , please add data.sql to the resources folder with the following script:

```

insert into employee (name) values ('pas');
insert into employee (name) values ('lucia');
insert into employee (name) values ('lucas');
insert into employee (name) values ('siena');

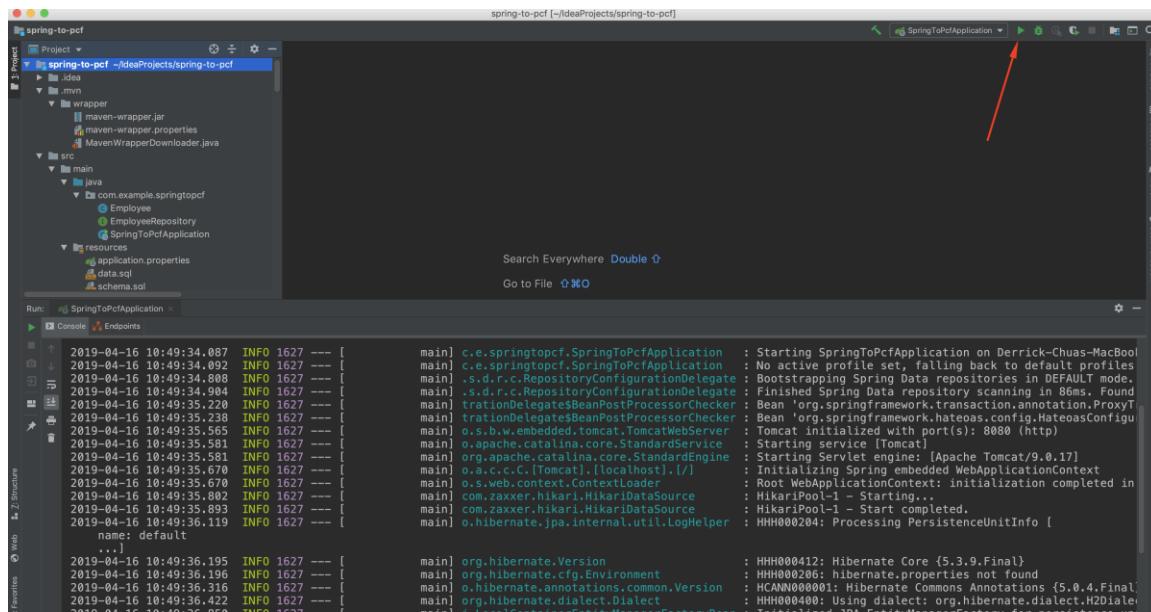
```

The screenshot shows the IntelliJ IDEA interface with the project 'SpringToPcf' open. The left sidebar displays the project structure with additional files: .gitignore and README.md. The right panel shows two tabs: 'schema.sql' and 'data.sql'. Both tabs contain the same SQL code as the previous screenshot, indicating that data.sql has been added to the resources folder.

- Now lets run our Spring Boot application by clicking on the green “Run” button as illustrated.

Few things to Note:

- Tomcat is bundled and used to run our application as per the spring starter pom.xml entries
- data.sql will be run once JPA has created the underlying table's in the database
- Various Spring Boot Auto Configurations have been run



The screenshot shows the IntelliJ IDEA interface with the project 'spring-to-pcf' open. The 'Run' toolbar at the top has a green play button icon for 'SpringToPcfApplication'. The code editor displays the main application class and its dependencies.

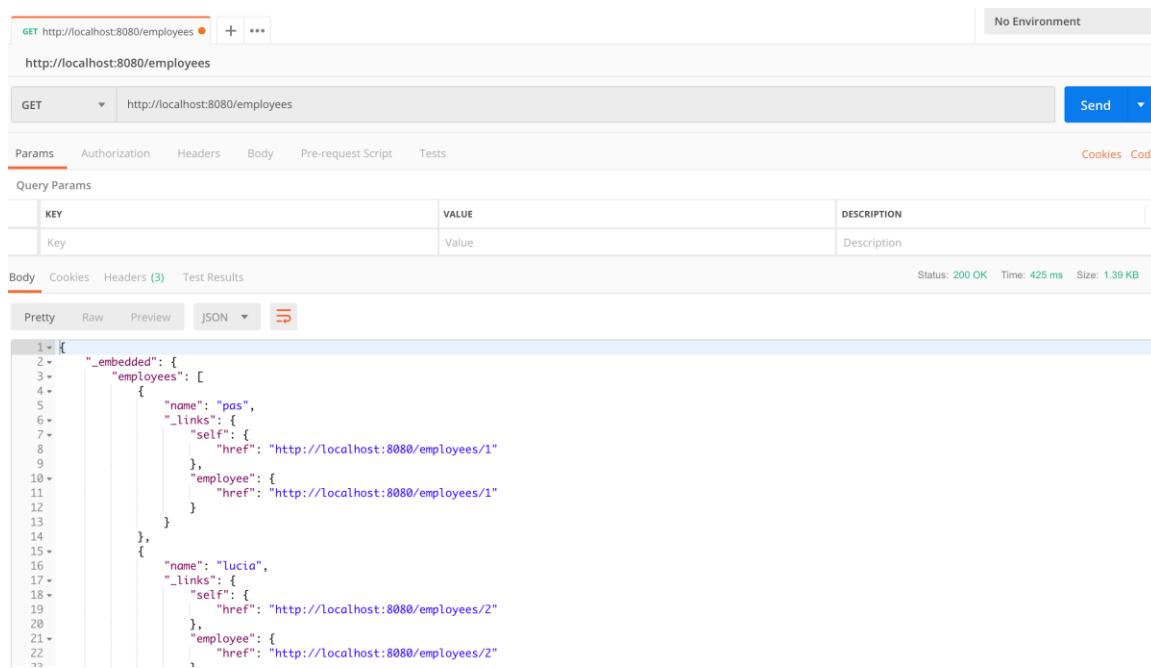
```

2019-04-16 10:49:34.087 INFO 1627 --- [           main] c.e.springtopcf.SpringToPcfApplication : Starting SpringToPcfApplication on Derrick-Chuas-MacBook-Pro.local with PID 1627
2019-04-16 10:49:34.092 INFO 1627 --- [           main] c.e.springtopcf.SpringToPcfApplication : No active profile set, falling back to default profiles
2019-04-16 10:49:34.984 INFO 1627 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data repositories in DEFAULT mode.
2019-04-16 10:49:35.220 INFO 1627 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 86ms. Found
2019-04-16 10:49:35.238 INFO 1627 --- [           main] .r.tionDelegates$BeanPostProcessorChecker : Bean 'org.springframework.transaction.annotation.ProxyT
2019-04-16 10:49:35.565 INFO 1627 --- [           main] .r.tionDelegates$BeanPostProcessorChecker : Bean 'org.springframework.transaction.annotation.ProxyT
2019-04-16 10:49:35.581 INFO 1627 --- [           main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2019-04-16 10:49:35.581 INFO 1627 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2019-04-16 10:49:35.670 INFO 1627 --- [           main] o.s.web.context.ContextLoader : Root WebApplicationContext: initialization completed in
2019-04-16 10:49:35.802 INFO 1627 --- [           main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...
2019-04-16 10:49:35.893 INFO 1627 --- [           main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.
2019-04-16 10:49:36.119 INFO 1627 --- [           main] o.hibernate.jpa.internal.util.LogHelper : HHH000204: Processing PersistenceUnitInfo [
    name: default
    ...]
2019-04-16 10:49:36.195 INFO 1627 --- [           main] org.hibernate.Version : HHH000412: Hibernate Core {5.3.9.Final}
2019-04-16 10:49:36.196 INFO 1627 --- [           main] org.hibernate.cfg.Environment : HHH000260: hibernate.properties not found
2019-04-16 10:49:36.316 INFO 1627 --- [           main] o.hibernate.annotations.common.Version : HCANN000001: Hibernate Commons Annotations {5.0.4.Final}
2019-04-16 10:49:36.422 INFO 1627 --- [           main] org.hibernate.dialect.Dialect : HHH000400: Using dialect: org.hibernate.dialect.H2Diale
2019-04-16 10:49:36.454 INFO 1627 --- [           main] org.hibernate.dialect.Dialect : HHH000400: Using dialect: org.hibernate.dialect.H2Diale

```

12. Your first Spring Boot microservice is now running. You can test this using Postman by executing a GET request on

<http://localhost:8080/employees>



The screenshot shows the Postman interface with a successful GET request to 'http://localhost:8080/employees'. The response body is a JSON array containing two employees:

```

1 [ {
2   "_embedded": {
3     "employees": [
4       {
5         "name": "pas",
6         "_links": {
7           "self": {
8             "href": "http://localhost:8080/employees/1"
9           },
10          "employee": {
11            "href": "http://localhost:8080/employees/1"
12          }
13        }
14      },
15      {
16        "name": "lucia",
17        "_links": {
18          "self": {
19            "href": "http://localhost:8080/employees/2"
20          },
21          "employee": {
22            "href": "http://localhost:8080/employees/2"
23          }
24        }
25      }
26    ]
27  }
28]

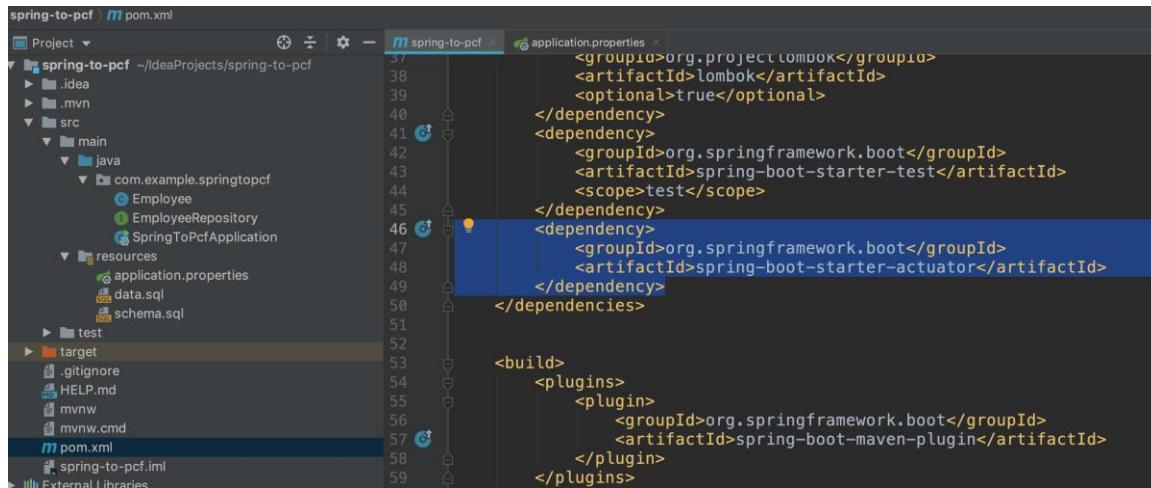
```

2. Add monitoring to your microservice

- Spring Boot Actuator is a sub-project of Spring Boot . It adds several Spring Boot's production-ready features to your application with little effort on your part.
You can find out more at
<https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/#production-ready>

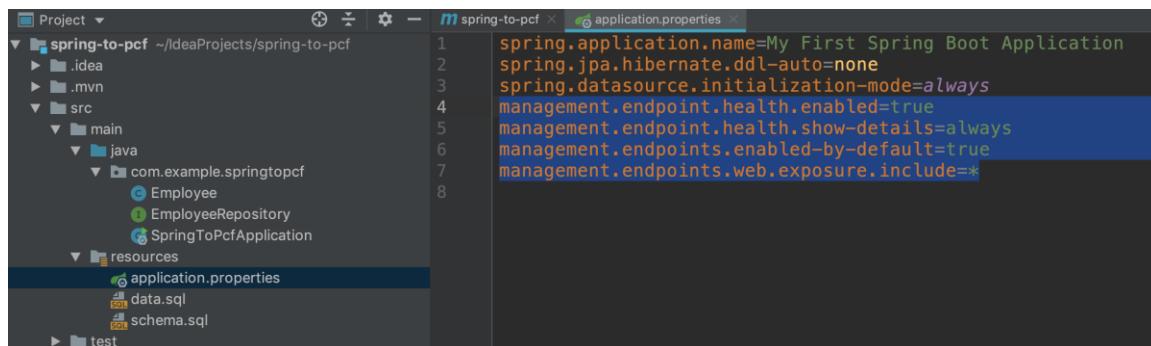
Now lets add the following dependency to the pom.xml of your microservice application.

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

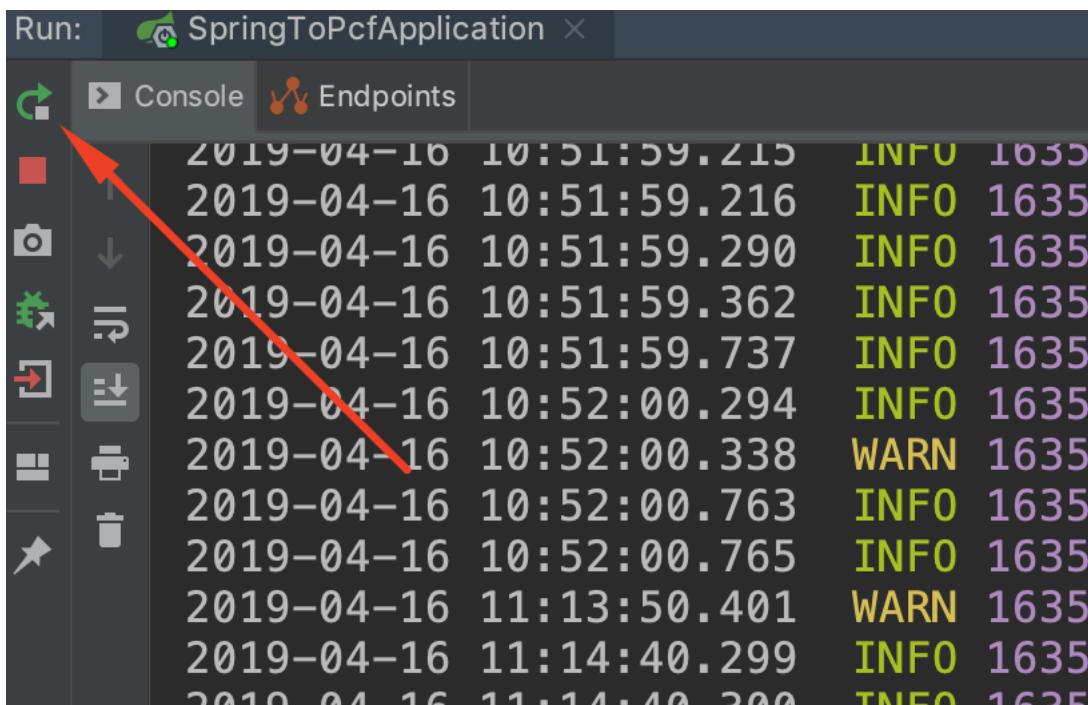


- We will also add some properties to enable the features. Please add the following to the application.properties:

```
management.endpoint.health.enabled=true
management.endpoint.health.show-details=always
management.endpoints.enabled-by-default=true
management.endpoints.web.exposure.include=*
```



3. Make sure the changes are saved and proceed to restart the application.



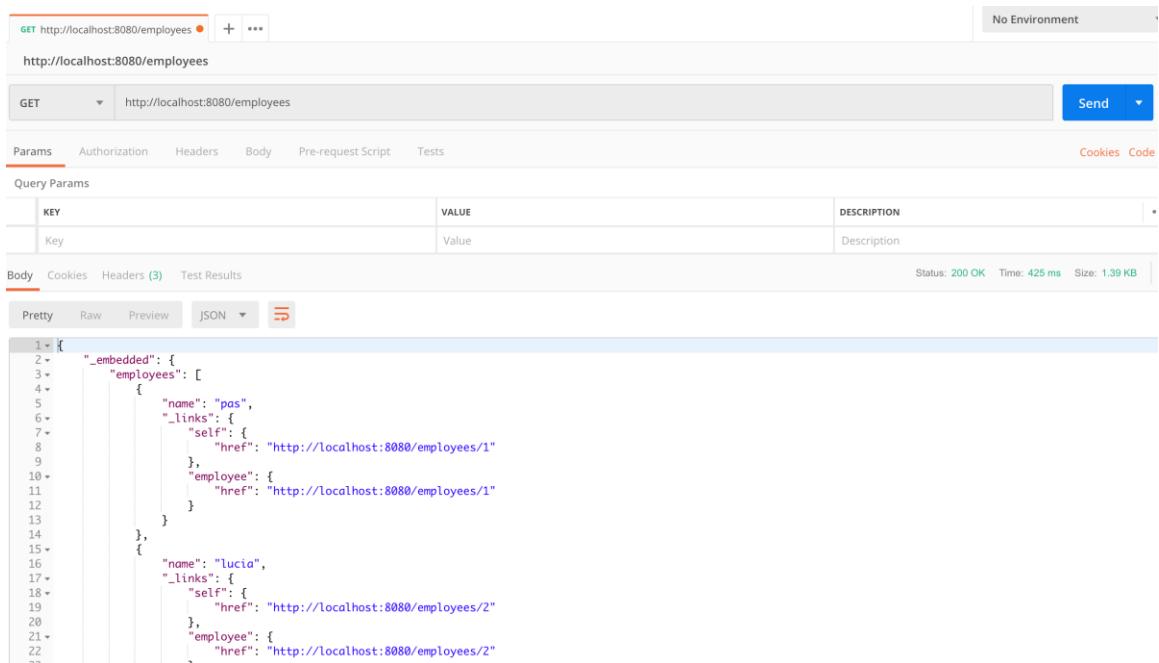
```

2019-04-16 10:51:59.215 INFO 1635
2019-04-16 10:51:59.216 INFO 1635
2019-04-16 10:51:59.290 INFO 1635
2019-04-16 10:51:59.362 INFO 1635
2019-04-16 10:51:59.737 INFO 1635
2019-04-16 10:52:00.294 INFO 1635
2019-04-16 10:52:00.338 WARN 1635
2019-04-16 10:52:00.763 INFO 1635
2019-04-16 10:52:00.765 INFO 1635
2019-04-16 11:13:50.401 WARN 1635
2019-04-16 11:14:40.299 INFO 1635
2019-04-16 11:14:40.300 INFO 1635

```

4. Ensure your employee microservice is still running. Execute the following HTTP request. You should see the same output.

GET http://localhost:8080/employees



KEY	VALUE	DESCRIPTION
Key	Value	Description

```

1: [
2:   "_embedded": {
3:     "employees": [
4:       {
5:         "name": "paul",
6:         "_links": {
7:           "self": {
8:             "href": "http://localhost:8080/employees/1"
9:           },
10:          "employee": {
11:            "href": "http://localhost:8080/employees/1"
12:          }
13:        }
14:      },
15:      {
16:        "name": "lucia",
17:        "_links": {
18:          "self": {
19:            "href": "http://localhost:8080/employees/2"
20:          },
21:          "employee": {
22:            "href": "http://localhost:8080/employees/2"
23:          }
24:        }
25:      }
26:    ]
27:  }
28: ]

```



5. Next execute the following HTTP request. You should get a list of actuator endpoints.

GET http://localhost:8080/actuator

```
1+ {
2+   "_links": {
3+     "self": {
4+       "href": "http://localhost:8080/actuator",
5+       "templated": false
6+     },
7+     "auditevents": {
8+       "href": "http://localhost:8080/actuator/auditevents",
9+       "templated": false
10+    },
11+   "beans": {
12+     "href": "http://localhost:8080/actuator/beans",
13+     "templated": false
14+   },
15+   "caches-cache": {
16+     "href": "http://localhost:8080/actuator/caches/{cache}",
17+     "templated": true
18+   },
19+   "caches": {
20+     "href": "http://localhost:8080/actuator/caches",
21+     "templated": false
22+   },
23+   "health": {
24+     "href": "http://localhost:8080/actuator/health",
25+     "templated": false
26+   }
27+ }
```

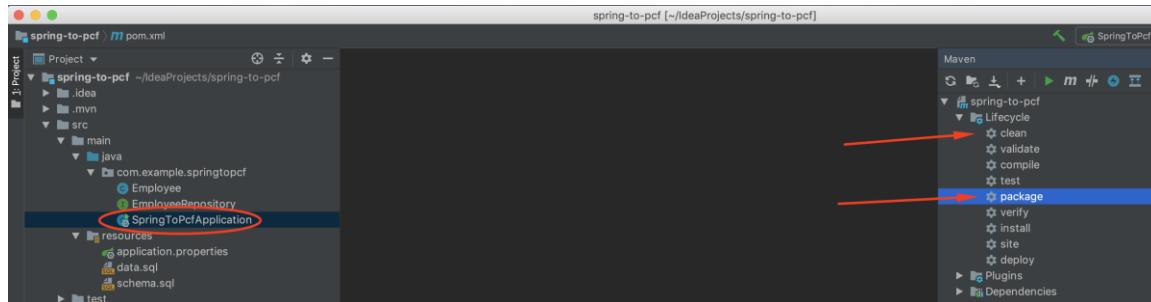
6. Try visiting the different health endpoint. You will see that the h2 database is in use.

GET http://localhost:8080/actuator/health

```
1+ {
2+   "status": "UP",
3+   "details": {
4+     "db": {
5+       "status": "UP",
6+       "details": {
7+         "database": "H2",
8+         "hello": 1
9+       }
10+     },
11+     "diskSpace": {
12+       "status": "UP",
13+       "details": {
14+         "total": 499963174912,
15+         "free": 286439755776,
16+         "threshold": 10485760
17+       }
18+     }
19+   }
20+ }
```

3. Deploying and running on Pivotal Application Service

1. Perform a maven clean and package of your microservice from the IDE. You can do so by double-clicking on each of the maven stages separately.



2. Open up a terminal and 'cd' to the project directory. Perform a 'ls' on the target directory and you should see the jar file.

```
1. bash
Derrick-Chua-MacBook-Pro:spring-to-pcf tmchua$ cd ~/IdeaProjects/spring-to-pcf/
Derrick-Chua-MacBook-Pro:spring-to-pcf tmchua$ ls target/
classes           maven-archiver
generated-sources   maven-status
generated-test-sources  spring-to-pcf-0.0.1-SNAPSHOT.jar
Derrick-Chua-MacBook-Pro:spring-to-pcf tmchua$
```

3. Using the cf cli, log in to your PWS account. The command to do so is

```
cf login -a https://api.run.pivotal.io
```

```
1. bash
Derrick-Chua-MacBook-Pro:target tmchua$ cf login -a https://api.run.pivotal.io
API endpoint: https://api.run.pivotal.io

Email> tmchua@tmchua.com
Password>
Authenticating...
OK

Select an org (or press enter to skip):
1. null
2. null

Org> 2
Targeted org null

Select a space (or press enter to skip):
1. development
2. ws-test

Space> 2
Targeted space ws-test

API endpoint: https://api.run.pivotal.io (API version: 2.133.0)
User: tmchua@tmchua.com
Org: null
Space: ws-test
```



4. Deploy and run the app on PWS using the following command:

```
cf push employee-api -p target/spring-to-pcf-0.0.1-SNAPSHOT.jar -i 1 -m 1g --random-route
```

```
Derrick-Chuas-MacBook-Pro:spring-to-pcf tmchua$ cf push employee-api -p target/spring-to-pcf-0.0.1-SNAPSHOT.jar -i 1 -m 1g --random-route
Pushing app employee-api to org tchua / space ws-test as tchua@pivotal.io...
Getting app info...
Creating app with these attributes...
+ name:          employee-api
+ path:          /Users/tmchua/IdeaProjects/spring-to-pcf/target/spring-to-pcf-0.0.1-SNAPSHOT.jar
+ instances:    1
+ memory:       1G
+ routes:
+   employee-api-rested-puku.cfapps.io

Creating app employee-api...
Mapping routes...
Comparing local files to remote cache...
Packaging files to upload...
Uploading files...
 400.72 KiB / 400.72 KiB [=====] 100.00% 6s

Waiting for API to complete processing files...
```

5. How do you now know the status of your app on the platform? You can do so through either the cf cli or through the PWS web console.

On the terminal, execute the following cf command:

```
cf app employee-api
```

```
Derrick-Chuas-MacBook-Pro:spring-to-pcf tmchua$ cf app employee-api
Showing health and status for app employee-api in org tchua / space ws-test as tchua@pivotal.io...

name:          employee-api
requested state: started
routes:        employee-api-rested-puku.cfapps.io
last uploaded: Tue 16 Apr 15:48:23 +08 2019
stack:         cflinuxfs3
buildpacks:    client-certificate-mapper=1.8.0_RELEASE container-security-provider=1.16.0_RELEASE
               java-buildpack=v4.19-offline-https://github.com/cloudfoundry/java-buildpack.git#3f4eee2 java-main java-opts java-security
               jvmkill-agent=1.16.0_RELEASE open-jdk=...

type:          web
instances:     1/1
memory usage: 1024M
state          since            cpu      memory      disk      details
+0  running    2019-04-16T07:49:07Z  0.4%  281M of 1G  148.4M of 1G
Derrick-Chuas-MACBOOK-Pro:spring-to-pcf tmchua$
```

On the browser, navigate to <https://console.run.pivotal.io> and log in if you have to. Click into your org and space, and then select the app. You should see the following. You can also observe that the actuator health endpoint has been integrated into the console.

Pivotal Web Services

Home / tchua / ws-test / employee-api

APP employee-api Running

Overview Services Route (1) Logs Tasks Trace Threads Settings

Events Last Push: 03:48 PM 04/16/19 App Summary

Started app tchua@pivotal.io 04/16/2019 at 03:48:05 PM

Mapped route to app tchua@pivotal.io 04/16/2019 at 03:47:36 PM

Created app tchua@pivotal.io 04/16/2019 at 03:47:32 PM

Instances / Allocated 1 / 1 Memory / Allocated 0.29 / 1.00 GB Disk / Allocated 0.14 / 1.00 GB

Processes and Instances

web

Instances 1 Memory Allocated 1 GB Disk Allocated 1 GB

Autoscaling

#	App Health	CPU	Memory	Disk	Uptime
0	Up	0%	293.3 MB	148.38 MB	19 min

Health Check

```

status: UP
db
  status: UP
  database: H2
  hello: 1
diskSpace
  status: UP
  free: 918159360
  threshold: 10485760

```

- From the status of 'cf app' command, you can see the route. This route forms the domain name part of your employee API URL. Now you can query your employee API on

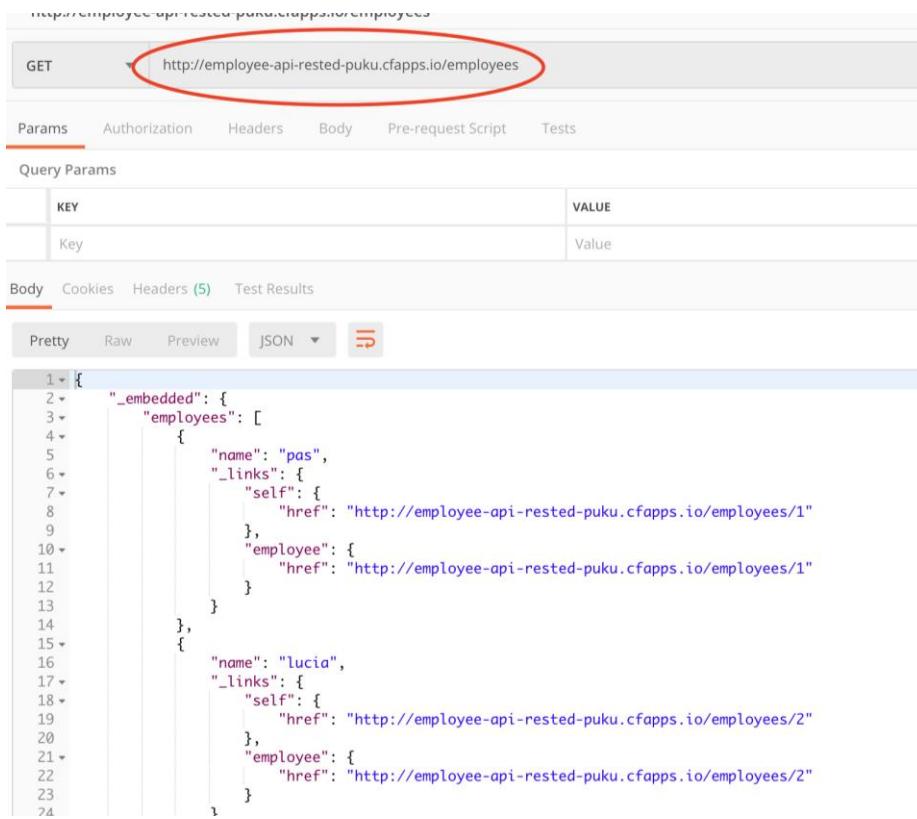
`http://{route}/employees`

You can try this on Postman.

```

Derrick-Chuas-MacBook-Pro:spring-to-pcf tmchua$ cf app employee-api
Showing health and status for app employee-api in org tchua / space ws-test as tchua@pivotal.io...
name: employee-api
requested state: started
routes: employee-api-rested-puku.cfapps.io
last uploaded: Tue Apr 16 15:48:23 +08 2019
stack: cflinuxfs3
buildpacks:
  client-certificate-mapper=1.8.0_RELEASE container-security-provider=1.16.0_RELEASE
  java-buildpack=v4.19-offline-https://github.com/cloudfoundry/java-buildpack.git#3f4eee2 java-main java-opts java-security
  jvmkill-agent=1.16.0_RELEASE open-jdk-...
type: web
instances: 1/1
memory usage: 1024M
  state since          cpu    memory      disk      details
#0  running 2019-04-16T07:49:07Z  0.4%  281M of 1G  148.4M of 1G
Derrick-Chuas-MacBook-Pro:spring-to-pcf tmchua$ 

```



http://employee-api-rested-puku.cfapps.io/employees

GET http://employee-api-rested-puku.cfapps.io/employees

Params Authorization Headers Body Pre-request Script Tests

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (5) Test Results

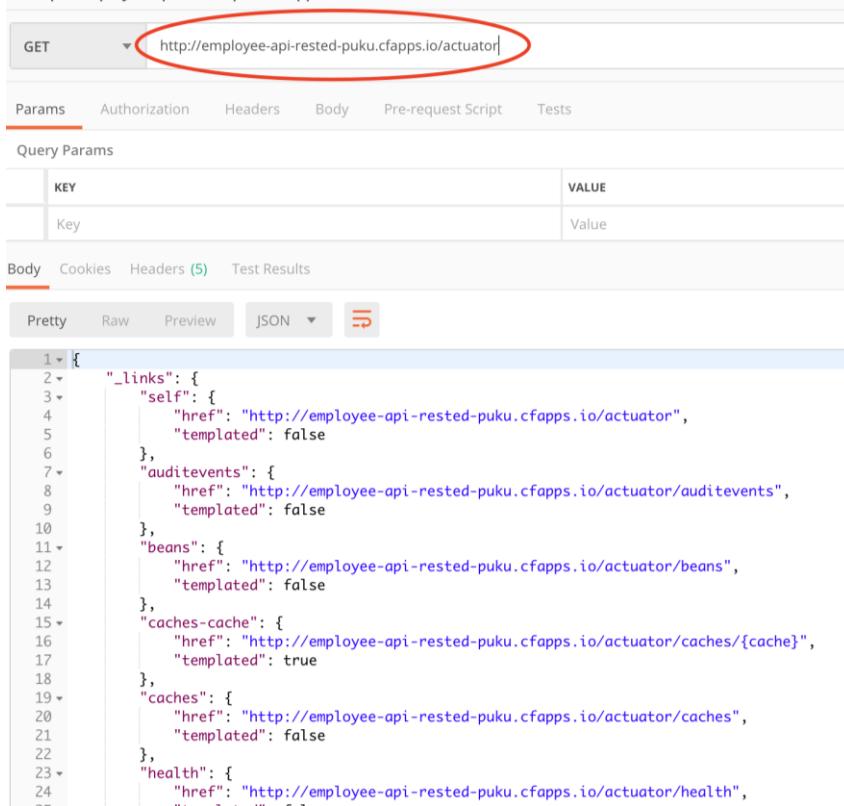
Pretty Raw Preview JSON

```

1 ↴ {
2   "_embedded": {
3     "employees": [
4       {
5         "name": "pas",
6         "_links": {
7           "self": {
8             "href": "http://employee-api-rested-puku.cfapps.io/employees/1"
9           },
10          "employee": {
11            "href": "http://employee-api-rested-puku.cfapps.io/employees/1"
12          }
13        },
14        {
15          "name": "lucia",
16          "_links": {
17            "self": {
18              "href": "http://employee-api-rested-puku.cfapps.io/employees/2"
19            },
20            "employee": {
21              "href": "http://employee-api-rested-puku.cfapps.io/employees/2"
22            }
23          }
24        }
      ]
    }
  }
}

```

7. You can also access the actuator endpoints the same way.



http://employee-api-rested-puku.cfapps.io/actuator

GET http://employee-api-rested-puku.cfapps.io/actuator

Params Authorization Headers Body Pre-request Script Tests

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (5) Test Results

Pretty Raw Preview JSON

```

1 ↴ {
2   "_links": {
3     "self": {
4       "href": "http://employee-api-rested-puku.cfapps.io/actuator",
5       "templated": false
6     },
7     "auditevents": {
8       "href": "http://employee-api-rested-puku.cfapps.io/actuator/auditevents",
9       "templated": false
10    },
11    "beans": {
12      "href": "http://employee-api-rested-puku.cfapps.io/actuator/beans",
13      "templated": false
14    },
15    "caches-cache": {
16      "href": "http://employee-api-rested-puku.cfapps.io/actuator/caches/{cache}",
17      "templated": true
18    },
19    "caches": {
20      "href": "http://employee-api-rested-puku.cfapps.io/actuator/caches",
21      "templated": false
22    },
23    "health": {
24      "href": "http://employee-api-rested-puku.cfapps.io/actuator/health",
25      "templated": false
26    }
  }
}

```

4. Creating and binding a service on demand

- In your PWS console, go to the “Services” tab. Click on “Add A Service” button.

Home / tchua / ws-test

SPACE	RUNNING	STOPPED	CRASHED
ws-test	● 1	● 0	● 0

App (1) Services Routes (2) Member (1) Settings

Services

ADD A SERVICE

Loading Services...

- In the panel that appears, search for ‘mysql’ and ClearDB should appear. Click on ClearDB, select the free Spark DB plan and then click “Select Plan”.

x Create a new service [View in Marketplace](#)

ClearDB MySQL Database
Highly available MySQL for your Apps.

x Create a new service [View in Marketplace](#)

ClearDB MySQL Database
Highly available MySQL for your Apps.

<input checked="" type="radio"/> Spark DB free Show Details
<input type="radio"/> Boost DB \$10.00/MONTH Show Details
<input type="radio"/> Amp DB \$50.00/MONTH Show Details
<input type="radio"/> Shock DB \$100.00/MONTH Show Details

BACK **SELECT PLAN**



3. Enter the value “pts-db” as the Instance Name. Ensure “Bind to App” field is set to “[do not bind]”. Proceed to click “Create”. Wait for a few minutes for the service to be created.

x Create a new service [View in Marketplace](#)

 ClearDB MySQL Database
spark - free

Instance Name
pts-db

Add to Space
ws-test

Bind to App (Optional)
[do not bind]

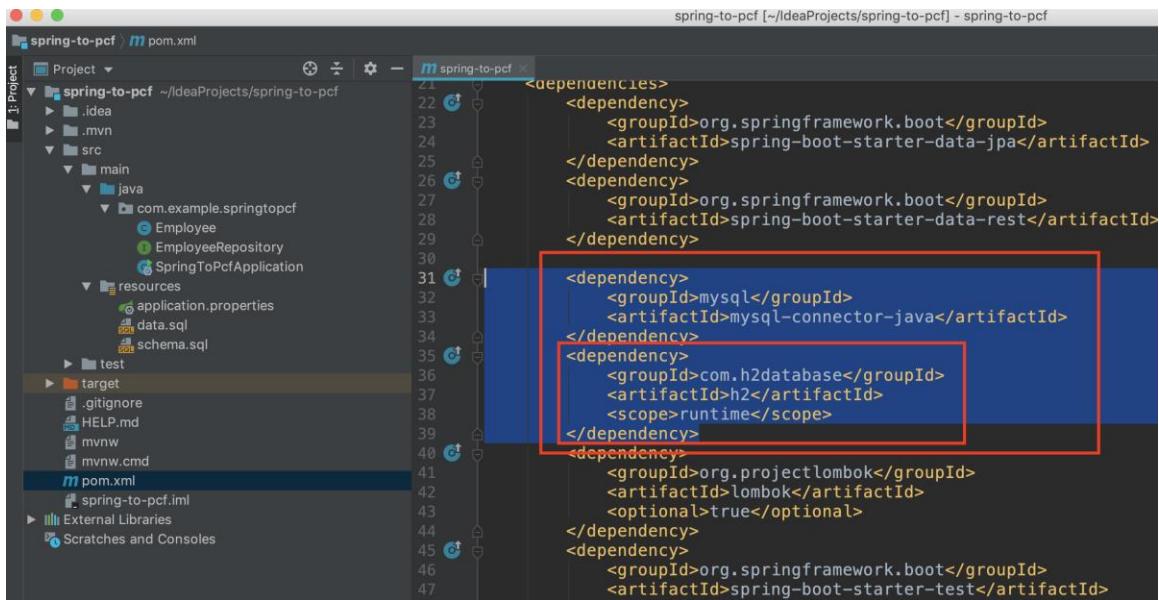
Add Parameters ⓘ (Optional) Enter JSON
Name Value +

Advanced Configuration ⓘ

[BACK](#) [CREATE](#)

4. With the MySQL instance created, now we need to make changes to the Employee API microservice to make use of MySQL, instead of h2. This will involve just a change in the pom.xml.
Add the following XML snippet to pom.xml, making sure that it is before the h2 dependency.

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



The screenshot shows the IntelliJ IDEA interface with the project 'spring-to-pcf' open. The 'pom.xml' file is the active editor. A red box highlights a section of the XML code where several database dependencies are listed:

```

<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
</dependency>
<dependency>
    <groupId>com.h2database</groupId>
    <artifactId>h2</artifactId>
    <scope>runtime</scope>
</dependency>
<dependency>
    <groupId>org.projectlombok</groupId>
    <artifactId>lombok</artifactId>
    <optional>true</optional>
</dependency>
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>

```

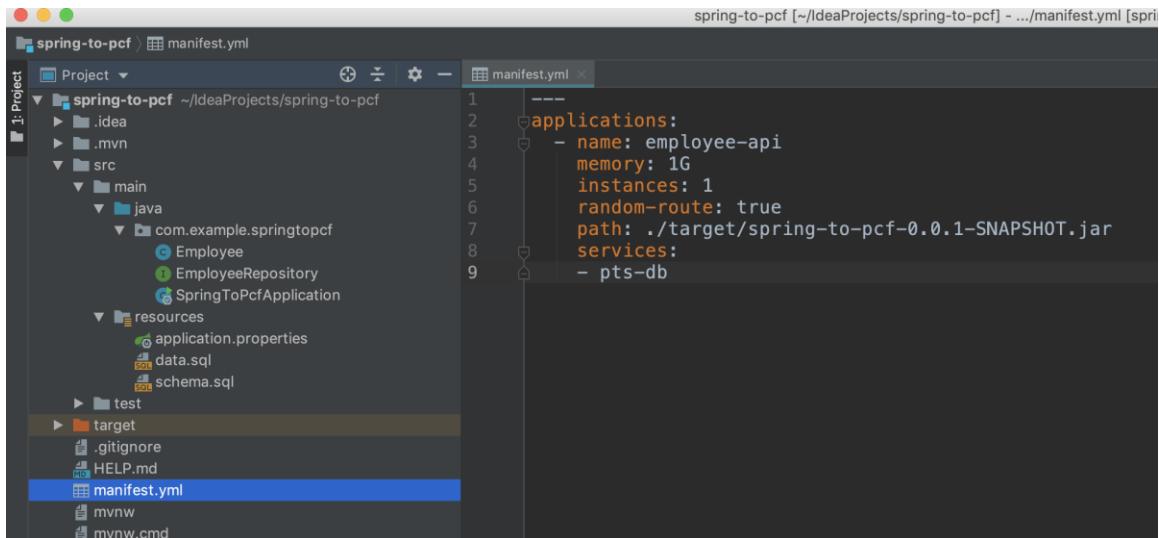
- Add a new file named "manifest.yml" to the project root. Include the following content in the file.

```

---
applications:
  - name: employee-api
    memory: 1G
    instances: 1
    random-route: true
    path: ./target/spring-to-pcf-0.0.1-SNAPSHOT.jar
    services:
      - pts-db

```

Instead of specifying arguments to the 'cf push' command, running 'cf push' without any argument will cause it to look for manifest.yml in the current working directory for parameters. Using a manifest will allow application specific parameters to be source controlled together with the application in git. It also does away with the need to configure and update CI/CD pipelines.



The screenshot shows the IntelliJ IDEA interface with the project 'spring-to-pcf' open. The 'manifest.yml' file is the active editor. The file contains the following YAML configuration:

```

---
applications:
  - name: employee-api
    memory: 1G
    instances: 1
    random-route: true
    path: ./target/spring-to-pcf-0.0.1-SNAPSHOT.jar
    services:
      - pts-db

```

Pivotal

- In the maven panel, double click on clean and package to create a new clean copy of the jar file. Switch over to the terminal and execute the command

```
cf push
```

You will see in the console log that the manifest file is being used to read in the app attributes

```
Derrick-Chuas-MacBook-Pro:spring-to-pcf tmchua$ cf push
Pushing from manifest to org tchua / space ws-test as tchua@pivotal.io...
Using manifest file /Users/tmchua/IdeaProjects/spring-to-pcf/manifest.yml
Getting app info...
Updating app with these attributes...
name: employee-api
path: /Users/tmchua/IdeaProjects/spring-to-pcf/target/spring-to-pcf-0.0.1-SNAPSHOT.jar
command: JAVA_OPTS="-agentpath:$PWD/.java-buildpack/open_jdk_jre/bin/jvmkill-1.16.0_RELEASE=printHeapHistogram=1 -Djava.io.tmpdir=$TMPDIR -XX:ActiveProcessorCount=$(nproc) -Djava.ext.dirs=$PWD/.java-buildpack/container_security_provider:$PWD/.java-buildpack/open_jdk_jre/lib/ext -Djava.security.properties=$PWD/.java-buildpack/java_security/java.security.$JAVA_OPTS" && CALCULATED_MEMORY=$(SPWD/.java-buildpack/open_jdk_jre/bin/java-buildpack-memory-calculator-3.13.0_RELEASE -totMemory=$MEMORY_LIMIT -loadedClasses=17737 -poolType=metaspaces -stackThreads=250 -vmOptions="$JAVA_OPTS") && echo JVM Memory Configuration: $CALCULATED_MEMORY && JAVA_OPTS="$JAVA_OPTS $CALCULATED_MEMORY" && MALLOC_ARENA_MAX=2 SERVER_PORT=$PORT eval exec $PWD/.java-buildpack/open_jdk_jre/bin/java $JAVA_OPTS -cp $PWD/.org.springframework.boot.loader.JarLauncher
disk quota: 1G
health check type: port
instances: 1
memory: 1G
stack: cflinuxfs3
services:
+ pts-db
routes:
employee-api-rested-puku.cfapps.io

Updating app employee-api...
Mapping routes...
Binding services...
Comparing local files to remote cache...
Packaging files to upload...
Uploading files...
400.75 KiB / 400.75 KiB [=====] 100.00% 3s
```

- Switch over to the console to look at the actuator based health status of the app, and you will see that it is now using MySQL. You might also want to verify that the Employee API is working correctly using Postman.

APP
employee-api Running

VIEW APP

Overview Service (1) Route (1) Logs Tasks Trace Threads Settings

Buildpack: N/A

Events Last Push: 06:07 PM 04/16/19 App Summary

Started app tchua@pivotal.io 04/16/2019 at 06:07:28 PM

Stopped app tchua@pivotal.io 04/16/2019 at 06:07:26 PM

Renamed app to employee-api tchua@pivotal.io 04/16/2019 at 06:07:08 PM

Started app tchua@pivotal.io 04/16/2019 at 03:48:05 PM

Mapped route to app tchua@pivotal.io 04/16/2019 at 03:47:36 PM

Created app tchua@pivotal.io 04/16/2019 at 03:47:32 PM

Processes and Instances

View in PCF Metrics

web

Instances	Memory Allocated	Disk Allocated
1	1 GB	1 GB

SCALE

Autoscaling

#	App Health	CPU	Memory	Disk	Uptime
0	Up	0%	261 MB	150.41 MB	1 min

Health Check

status: UP
db: status: UP
database: MySQL
hello: 1
diskSpace:
status: UP
free: 916021248
threshold: 10485760
total: 1073741824

VIEW JSON

The screenshot shows the Postman application interface. At the top, there is a header bar with a 'GET' button, a URL input field containing 'http://employee-api-rested-puku.cfapps.io/employees', and a '+' and '...' button. Below the header is a navigation bar with tabs: 'GET', 'http://employee-api-rested-puku.cfapps.io/employees', 'Params', 'Authorization', 'Headers', 'Body', 'Pre-request Script', and 'Tests'. The 'Params' tab is currently selected. Under 'Query Params', there is a table with one row: 'Key' and 'Value'. Below the navigation bar is another set of tabs: 'Body', 'Cookies', 'Headers (5)', and 'Test Results', with 'Body' being the active tab. Under 'Body', there are three options: 'Pretty', 'Raw', and 'JSON'. The 'JSON' option is selected and has a dropdown menu with a 'Copy' icon. The main content area displays a JSON response with line numbers from 1 to 30 on the left. The JSON structure is as follows:

```

1  {
2    "_embedded": {
3      "employees": [
4        {
5          "name": "pas",
6          "_links": {
7            "self": {
8              "href": "http://employee-api-rested-puku.cfapps.io/employees/2"
9            },
10           "employee": {
11             "href": "http://employee-api-rested-puku.cfapps.io/employees/2"
12           }
13         },
14       },
15       {
16         "name": "lucia",
17         "_links": {
18           "self": {
19             "href": "http://employee-api-rested-puku.cfapps.io/employees/12"
20           },
21           "employee": {
22             "href": "http://employee-api-rested-puku.cfapps.io/employees/12"
23           }
24         },
25       },
26       {
27         "name": "lucas",
28         "_links": {
29           "self": {
30             "href": "http://employee-api-rested-puku.cfapps.io/employees/22"
31           }
32         }
33       }
34     ]
35   }
36 }
```

5. Scaling your app

1. Navigate to the PWS console and go to your employee-api app. Click on the “Scale” button as shown.

Home / tchua / ws-test / employee-api

APP
employee-api Running [VIEW APP](#)

Overview Service (1) Route (1) Logs Tasks Trace Threads Settings Buildpack: N/A

Events Last Push: 06:07 PM 04/16/19 App Summary

Started app tchua@pivotalko 04/16/2019 at 06:07:28 PM	Instances / Allocated 1 / 1	Memory / Allocated 0.27 / 1.00 GB	Disk / Allocated 0.15 / 1.00 GB
Stopped app tchua@pivotalko 04/16/2019 at 06:07:26 PM			
Renamed app to employee-api tchua@pivotalko 04/16/2019 at 06:07:08 PM			
Started app tchua@pivotalko 04/16/2019 at 03:48:05 PM			
Mapped route to app tchua@pivotalko 04/16/2019 at 03:47:36 PM			
Created app tchua@pivotalko 04/16/2019 at 03:47:32 PM			

Processes and Instances [View in PCF Metrics](#)

web

Instances	Memory Allocated	Disk Allocated	SCALE
<input checked="" type="checkbox"/> Autoscaling			
#	App Health	CPU	Memory
> 0	Up	0%	278.98 MB
			150.41 MB
			2 hr 51 min
			:

2. Increase the number of instances to 2 and click on “Apply Changes”

x Scale app

web

Instances	Memory Limit	Disk Limit
<input type="text" value="2"/>	1 GB	1 GB

Usage Total

Instances	Memory Limit	Disk Limit
2	2.00 GB	2.00 GB

APPLY CHANGES

3. Observe that action has been taken to scale the app instances to 2.

Home / tchua / ws-test / employee-api

- Switch over to your terminal and execute the following command:

```
cf logs employee-api
```

This will activate the log reading functionality in tail mode

```
1. cf
Derrick-Chuas-MacBook-Pro:spring-to-pcf tmchua$ cf logs employee-api
Retrieving logs for app employee-api in org tchua / space ws-test as tchua@pivotal.io...
|
```

- Using Postman, execute this HTTP request multiple times to generate some logs:

```
http://{route}/employees
```

Observe the log output at the terminal, looking specifically for a json property called "app_index". You should see values "0" and "1" interspersed.

```

1. cf
t sent successfully to the server was 126,344 milliseconds ago.). Possibly consider using a shorter maxLifetime value.
2019-04-16T21:16:13.40+0800 [APP/PROC/WEB/0] OUT 2019-04-16 13:16:13.402 INFO 15 --- [connection adder] ContainerTrustManagerFactory$PKIXFactory : Adding System Trust Manager
2019-04-16T21:16:13.56+0800 [RTR/7] OUT employee-api-rested-puku.cfapps.io - [2019-04-16T13:16:13.283+0000] "GET /employees HTTP/1.1" 200 0 1506 "-" "PostmanRuntime/7.6.1" "10.10.66.182:1262" "10.10.149.165:61048" x_forwarded_for:"132.147.124.222, 10.10.66.182" x_forwarded_proto:"https" vcap_request_id:"10811889-d594-4cc3-488d-126e250f3f7" response_time:0.281641509 app_id:"a305248e-877a-48ea-a0ef-37cba5ccc632" app_index:"0" x_b3_traceid:"9c25d641b945b924" x_b3_spanid:"9c25d641b945b924" x_b3_parentspanid:"-" b3:"9c25d641b945b924"
2019-04-16T21:16:13.56+0800 [RTR/7] OUT
2019-04-16T21:16:14.87+0800 [RTR/7] OUT employee-api-rested-puku.cfapps.io - [2019-04-16T13:16:14.733+0000] "GET /employees HTTP/1.1" 200 0 1506 "-" "PostmanRuntime/7.6.1" "10.10.66.182:45316" "10.10.149.165:61048" x_forwarded_for:"132.147.124.222, 10.10.66.182" x_forwarded_proto:"https" vcap_request_id:"44a1ff82-f65c-4762-68da-3f75a1d67f50" response_time:0.145364885 app_id:"a305248e-877a-48ea-a0ef-37cba5ccc632" app_index:"0" x_b3_traceid:"830f770fe1d87851" x_b3_spanid:"830f770fe1d87851" x_b3_parentspanid:"-" b3:"830f770fe1d87851-830f770fe1d87851"
2019-04-16T21:16:14.87+0800 [RTR/9] OUT
2019-04-16T21:16:15.75+0800 [APP/PROC/WEB/1] OUT 2019-04-16 13:16:15.758 WARN 26 --- [io-8080-exec-10] com.zaxxer.hikari.pool.PoolBase : HikariPool-1 - Failed to validate connection.com.mysql.cj.jdbc.ConnectionImpl@3f0e07b4 (Communications link failure)
2019-04-16T21:16:15.75+0800 [APP/PROC/WEB/1] OUT The last packet successfully received from the server was 128,366 milliseconds ago. The last packet sent successfully to the server was 128,367 milliseconds ago.). Possibly consider using a shorter maxLifetime value.
2019-04-16T21:16:15.78+0800 [APP/PROC/WEB/1] OUT 2019-04-16 13:16:15.788 INFO 26 --- [connection adder] ContainerTrustManagerFactory$PKIXFactory : Adding System Trust Manager
2019-04-16T21:16:15.83+0800 [RTR/5] OUT employee-api-rested-puku.cfapps.io - [2019-04-16T13:16:15.735+0000] "GET /employees HTTP/1.1" 200 0 1506 "-" "PostmanRuntime/7.6.1" "10.10.66.182:46602" "10.10.148.147:61218" x_forwarded_for:"132.147.124.222, 10.10.66.182" x_forwarded_proto:"https" vcap_request_id:"7a8dd1a7-a005-47c2-6b8a-ac3b73dc8460" response_time:0.096399273 app_id:"a305248e-877a-48ea-a0ef-37cba5ccc632" app_index:"0" x_b3_traceid:"ff182e058a79e3c4" x_b3_spanid:"ff182e058a79e3c4" x_b3_parentspanid:"-" b3:"ff182e058a79e3c4-ff182e058a79e3c4"
2019-04-16T21:16:15.83+0800 [RTR/5] OUT
2019-04-16T21:16:16.78+0800 [RTR/6] OUT employee-api-rested-puku.cfapps.io - [2019-04-16T13:16:16.743+0000] "GET /employees HTTP/1.1" 200 0 1506 "-" "PostmanRuntime/7.6.1" "10.10.66.182:3458" "10.10.149.165:61048" x_forwarded_for:"132.147.124.222, 10.10.66.182" x_forwarded_proto:"https" vcap_request_id:"f9b1514f-099a-4009-5b48-220bb0ba1a22" response_time:0.042835298 app_id:"a305248e-877a-48ea-a0ef-37cba5ccc632" app_index:"0" x_b3_traceid:"29fb8ef545de7348" x_b3_spanid:"29fb8ef545de7348" x_b3_parentspanid:"-" b3:"29fb8ef545de7348-29fb8ef545de7348"
2019-04-16T21:16:16.78+0800 [RTR/6] OUT
2019-04-16T21:16:17.58+0800 [RTR/11] OUT employee-api-rested-puku.cfapps.io - [2019-04-16T13:16:17.529+0000] "GET /employees HTTP/1.1" 200 0 1506 "-" "PostmanRuntime/7.6.1" "10.10.66.182:4750" "10.10.149.165:61048" x_forwarded_for:"132.147.124.222, 10.10.66.182" x_forwarded_proto:"https" vcap_request_id:"c76892ee-89a4-49d5-691f-1b82f944c608" response_time:0.053679753 app_id:"a305248e-877a-48ea-a0ef-37cba5ccc632" app_index:"0" x_b3_traceid:"bd4ddc2f8b253332" x_b3_spanid:"bd4ddc2f8b253332" x_b3_parentspanid:"-" b3:"bd4ddc2f8b253332-bd4ddc2f8b253332"
2019-04-16T21:16:17.58+0800 [RTR/11] OUT

```

6. Reduce the number of instances back to 1, execute the GET request a few times and observe the log output. You should find that only instance 0 is handling the requests.

Home / tchua / ws-test / employee-api

APP

employee-api ● Running

VIEW APP

Overview	Service (1)	Route (1)	Logs	Tasks	Trace	Threads	Settings	Buildpack: N/A																											
Events	Last Push: 09:18 PM 04/16/19 App Summary																																		
Scaled app instances to 1 tchua@pivotal.io 04/16/2019 at 09:18:36 PM	Instances / Allocated	Memory / Allocated	Disk / Allocated																																
	1 / 1	0.28 / 1.00 GB	0.15 / 1.00 GB																																
Processes and Instances	View in PCF Metrics																																		
web <table border="1"> <thead> <tr> <th>Instances</th> <th>Memory Allocated</th> <th>Disk Allocated</th> <th>SCALE</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>1 GB</td> <td>1 GB</td> <td>SCALE</td> </tr> <tr> <td colspan="4"> <input checked="" type="checkbox"/> Autoscaling </td> </tr> <tr> <th>#</th> <th>App Health</th> <th>CPU</th> <th>Memory</th> <th>Disk</th> <th>Uptime</th> <td colspan="2"></td> </tr> <tr> <td>></td> <td>0</td> <td>Up</td> <td>3%</td> <td>286.58 MB</td> <td>150.41 MB</td> <td>3 hr 8 min</td> <td>⋮</td> </tr> </tbody> </table>								Instances	Memory Allocated	Disk Allocated	SCALE	1	1 GB	1 GB	SCALE	<input checked="" type="checkbox"/> Autoscaling				#	App Health	CPU	Memory	Disk	Uptime			>	0	Up	3%	286.58 MB	150.41 MB	3 hr 8 min	⋮
Instances	Memory Allocated	Disk Allocated	SCALE																																
1	1 GB	1 GB	SCALE																																
<input checked="" type="checkbox"/> Autoscaling																																			
#	App Health	CPU	Memory	Disk	Uptime																														
>	0	Up	3%	286.58 MB	150.41 MB	3 hr 8 min	⋮																												

Events

Last Push: 09:18 PM 04/16/19 App Summary

Scaled app instances to 1
tchua@pivotal.io 04/16/2019 at 09:18:36 PM

Scaled app instances to 2
tchua@pivotal.io 04/16/2019 at 09:05:06 PM

Started app
tchua@pivotal.io 04/16/2019 at 06:07:28 PM

Stopped app
tchua@pivotal.io 04/16/2019 at 06:07:26 PM

Renamed app to employee-api
tchua@pivotal.io 04/16/2019 at 06:07:08 PM

Started app
tchua@pivotal.io 04/16/2019 at 03:48:05 PM

Processes and Instances

View in PCF Metrics

web

Instances 1 Memory Allocated 1 GB Disk Allocated 1 GB SCALE

Autoscaling

#	App Health	CPU	Memory	Disk	Uptime		
>	0	Up	3%	286.58 MB	150.41 MB	3 hr 8 min	⋮

```

1. cf
10.2.23:59182" "10.10.149.165:61048" x_forwarded_for:"132.147.124.222, 10.10.2.23" x_forwarded_proto:"https" vcap_request_id:"dc7bdccfc-3738-4b24-4d05-b
664f927a915" response_time:0.012442837 app_id:"a305248e-877a-48ea-a0ef-37cba5ccc632" app_index:"0" x_b3_traceid:"5f51be7fe821916d" x_b3_spanid:"5f51be7
fe821916d" x_b3_parentspanid:"" b3:"5f51be7fe821916d" x_b3_traceid:"5f51be7fe821916d" x_b3_spanid:"5f51be7
2019-04-16T21:19:57.77+0800 [RTR/3] OUT
2019-04-16T21:19:57.76+0800 [RTR/1] OUT employee-api-rested-puku.cfapps.io - [2019-04-16T13:19:57.757+0000] "OPTIONS /cloudfoundryapplication/info HTTP/1.1" 200 0 0 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_2) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.103 Safari/537.36" "10.10.16.182:4912" "10.10.149.165:61048" x_forwarded_for:"132.147.124.222, 10.10.66.182" x_forwarded_proto:"https" vcap_request_id:"6a0ad99-cb00-4d2c-5d43-05aa8c312981" response_time:0.005311171 app_id:"a305248e-877a-48ea-a0ef-37cba5ccc632" app_index:"0" x_b3_traceid:"59cb5725330fe18c" x_b3_spanid:"59cb57
25330fe18c" x_b3_parentspanid:"" b3:"59cb5725330fe18c-59cb5725330fe18c"
2019-04-16T21:19:57.76+0800 [RTR/1] OUT
2019-04-16T21:19:57.76+0800 [RTR/0] OUT employee-api-rested-puku.cfapps.io - [2019-04-16T13:19:57.757+0000] "OPTIONS /cloudfoundryapplication/mappings HTTP/1.1" 200 0 0 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_2) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.103 Safari/537.36" "10.10.2.23:56706" "10.10.149.165:61048" x_forwarded_for:"132.147.124.222, 10.10.2.23" x_forwarded_proto:"https" vcap_request_id:"2c9cdad9-1319-48d7-7c7a-36b0d4dd3294" response_time:0.005081656 app_id:"a305248e-877a-48ea-a0ef-37cba5ccc632" app_index:"0" x_b3_traceid:"f53d09259a3e5961" x_b3_spanid:"f53d0
9259a3e5961" x_b3_parentspanid:"" b3:"f53d09259a3e5961-f53d09259a3e5961"
2019-04-16T21:19:57.76+0800 [RTR/0] OUT
2019-04-16T21:19:58.08+0800 [RTR/9] OUT employee-api-rested-puku.cfapps.io - [2019-04-16T13:19:58.009+0000] "GET /cloudfoundryapplication/info HTTP/1.1" 200 0 2 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_2) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.103 Safari/537.36" "10.10.182:46214" "10.10.149.165:61048" x_forwarded_for:"132.147.124.222, 10.10.66.182" x_forwarded_proto:"https" vcap_request_id:"6272c66-962c-4147-6e8f-c56
9f7669ac8" response_time:0.079347061 app_id:"a305248e-877a-48ea-a0ef-37cba5ccc632" app_index:"0" x_b3_traceid:"c5ce2db346c527ce" x_b3_spanid:"c5ce2db34
6c527ce" x_b3_parentspanid:"" b3:"c5ce2db346c527ce-c5ce2db346c527ce"
2019-04-16T21:19:58.08+0800 [RTR/9] OUT
2019-04-16T21:19:58.09+0800 [RTR/7] OUT employee-api-rested-puku.cfapps.io - [2019-04-16T13:19:58.012+0000] "GET /cloudfoundryapplication/health HTTP/1.1" 200 0 186 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_2) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.103 Safari/537.36" "10.10.2.23:59956" "10.10.149.165:61048" x_forwarded_for:"132.147.124.222, 10.10.2.23" x_forwarded_proto:"https" vcap_request_id:"47e6b577-1348-44fc-5259-23e
24169c5" response_time:0.077969907 app_id:"a305248e-877a-48ea-a0ef-37cba5ccc632" app_index:"0" x_b3_traceid:"519c09db7b5dcdba" x_b3_spanid:"519c09db7
b5dcdba" x_b3_parentspanid:"" b3:"519c09db7b5dcdba-519c09db7b5dcdba"
2019-04-16T21:19:58.09+0800 [RTR/7] OUT
2019-04-16T21:19:58.09+0800 [RTR/2] OUT employee-api-rested-puku.cfapps.io - [2019-04-16T13:19:57.998+0000] "GET /cloudfoundryapplication/mappings HTTP/1.1" 200 0 73888 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_2) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.103 Safari/537.36" "10.10.2.23:9184" "10.10.149.165:61048" x_forwarded_for:"132.147.124.222, 10.10.2.23" x_forwarded_proto:"https" vcap_request_id:"edc15518-929f-46c2-70ca
-f71f3e23197b" response_time:0.097891671 app_id:"a305248e-877a-48ea-a0ef-37cba5ccc632" app_index:"0" x_b3_traceid:"36c862f20b8248d6" x_b3_spanid:"36c862
f20b8248d6" x_b3_parentspanid:"" b3:"36c862f20b8248d6-36c862f20b8248d6"
2019-04-16T21:19:58.09+0800 [RTR/2] OUT

```

7. Enable “Autoscaling” and then restage your app using the restage button.

Success: Service instance "autoscale-ws-test" created and successfully bound to "employee-api". TIP: Restage your app to ensure your service binding is available to your app.

Home / tchua / ws-test / employee-api

APP employee-api Starting VIEW APP

Overview Services (2) Route (1) Logs Tasks Trace Threads Settings Buildpack: N/A

Events Last Push: 09:21 PM 04/16/19 App Summary

	Instances / Allocated	Memory / Allocated	Disk / Allocated
	1 / 2	0.28 / 2.00 GB	0.15 / 2.00 GB

Processes and Instances View in PCF Metrics

web	Instances	Memory Allocated	Disk Allocated	SCALE	
<input checked="" type="checkbox"/> Autoscaling	2	1 GB	1 GB	SCALE	
	App Health	CPU	Memory	Disk	Uptime
> 0	Up	0%	287.51 MB	150.41 MB	3 hr 11 min
> 1	STARTING...				

8. Click on “Manage Autoscaling”

web

Instances	Memory Allocated	Disk Allocated	SCALE
2	1 GB	1 GB	SCALE

Autoscaling MANAGE AUTOSCALING

#	App Health	CPU	Memory	Disk	Uptime
> 0	STARTING...				
> 1	STARTING...				

Pivotal

9. Add a scaling rule, starting by clicking on the “Edit” link. Add a CPU utilization rule as shown below. Note that the numbers used are for demo purposes only. In actual deployments, more sensible numbers should be used.

- × Manage Autoscaling

[Download Autoscaler CLI](#)

INSTANCE LIMITS
Currently in use: 2

Minimum Maximum **APPLY CHANGES**

SCALING RULES EDIT

No rules set

SCHEDULED LIMITS EDIT

◀ [Edit Scaling Rules](#)

Apps scale by 1 instance per event. Apps will scale up when any metric maximum is met and scale down only when all metric minimums are met.

CPU Utilization ▼

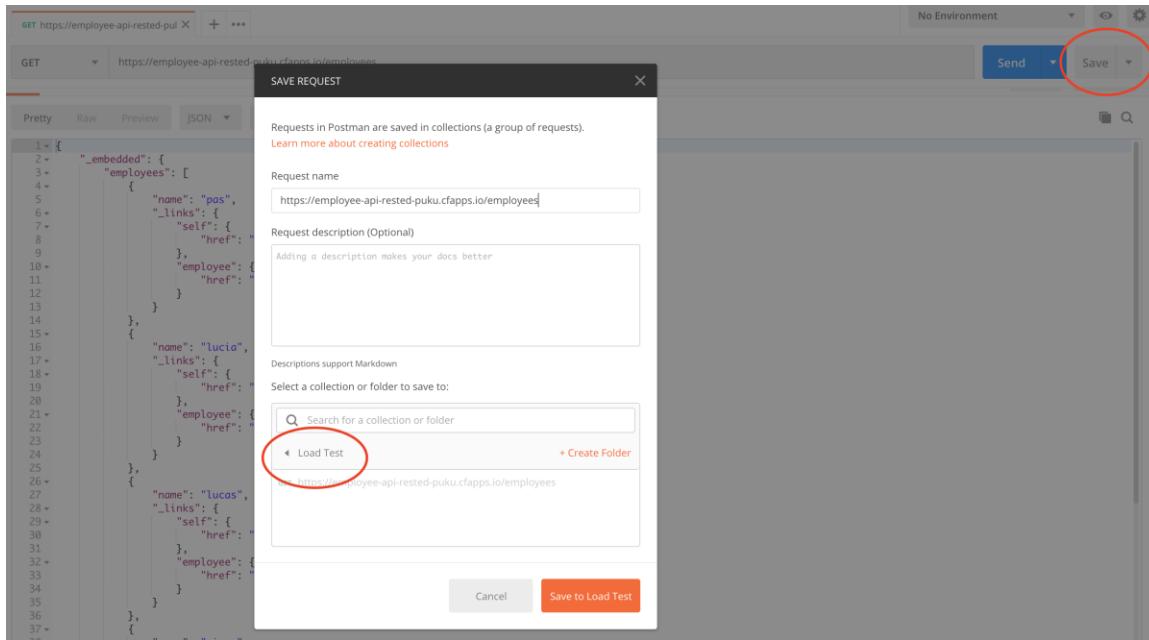
Scale down if less than: %

Scale up if more than: %

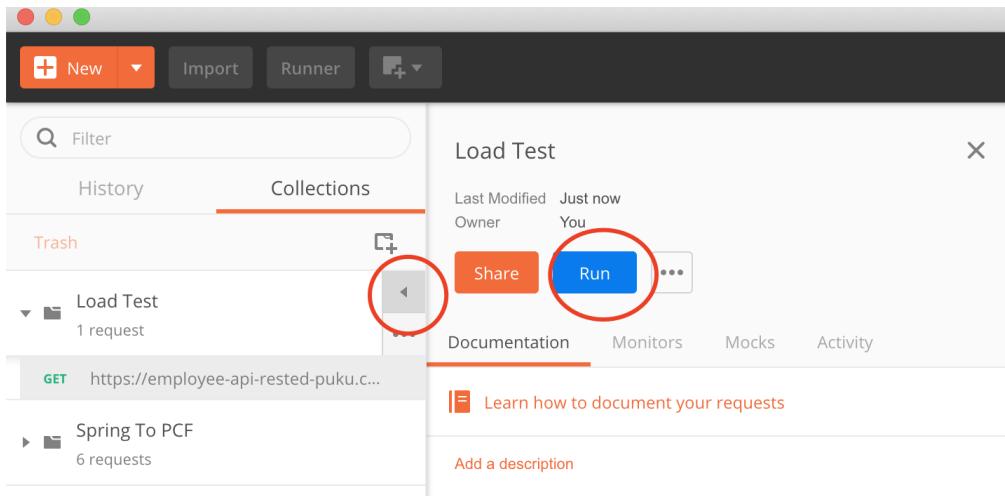
ADD RULE

CANCEL **SAVE**

10. In Postman, save your employees request to a new collection called “Load Test”.



- On the left panel, expand the arrow next to "Load Test" collection and click on the "Run" button.



- In the window that pops up, set the number of Iterations to 500, the Log Responses to "For no requests" and then hit the "Run Load Test" button.

Pivotal.

Choose a collection or folder

Search for a collection or folder

◀ Load Test

GET https://employee-api-rested-puku.cfapps.io/employees

Environment No Environment ▾

Iterations 500

Delay 0 ms

Log Responses For no requests ▾

Data Select File

Keep variable values ⓘ

Run Load Test

13. Switch back to the PWS console and observe your app. You should see the autoscaling kick in and spin up the third instance. You will also receive a notification in your email inbox.

Note: You might find the app crashing after some time. This is normal for this test. The app crashes due to the connections to the database running out because we are running on the free plan.



Home / tchua / ws-test / employee-api

APP employee-api [] [] [] ● Starting

VIEW APP []

Overview Services (2) Route (1) Logs Tasks Trace Threads Settings Buildpack: N/A

Events Last Push: 09:34 PM 04/16/19 App Summary

Updated app 04/16/2019 at 09:34:48 PM	Instances / Allocated 2 / 3	Memory / Allocated 0.76 / 3.00 GB	Disk / Allocated 0.44 / 3.00 GB
--	--------------------------------	--------------------------------------	------------------------------------

Processes and Instances

View in PCF Metrics []

web

Instances 3	Memory Allocated 1 GB	Disk Allocated 1 GB	SCALE		
Autoscaling					
#	App Health	CPU	Memory	Disk	Uptime
> 0	Up	14%	305.93 MB	150.41 MB	10 min
> 1	Up	3%	299.93 MB	150.41 MB	11 min
> 2	STARTING...				

MANAGE AUTOSCALING []



A scaling event has been reported for application **employee-api**.

Event Message: Scaled up from 2 to 3 instances. Current CPU of 6.87% is above upper threshold of 5.00%.

Event Time: 2019-04-16 13:34:48 +0000 UTC

The autoscaler will continue to scale this app within the current instance limits:

Minimum Instances: 2

Maximum Instances: 3

- After the tests completes, observe a while and you should see the number of instances dropping back to 2. An email notification will also be sent to you.

Home / tchua / ws-test / employee-api

APP employee-api [] [] [] ● Running

VIEW APP []

Overview Services (2) Route (1) Logs Tasks Trace Threads Settings Buildpack: N/A

Events Last Push: 09:40 PM 04/16/19 App Summary

Updated app 04/16/2019 at 09:40:04 PM	Instances / Allocated 2 / 2	Memory / Allocated 0.60 / 2.00 GB	Disk / Allocated 0.29 / 2.00 GB
--	--------------------------------	--------------------------------------	------------------------------------

Processes and Instances

View in PCF Metrics []

web

Instances 2	Memory Allocated 1 GB	Disk Allocated 1 GB	SCALE		
Autoscaling					
#	App Health	CPU	Memory	Disk	Uptime
> 0	Up	0%	308.03 MB	150.41 MB	15 min
> 1	Up	0%	301.86 MB	150.41 MB	16 min

MANAGE AUTOSCALING []



CF Notification: Scaling Down [Inbox](#)

no-reply@run.pivotal.io
to me ▾

9:40 PM (50 minutes ago)

Pivotal CF

A scaling event has been reported for application **employee-api**.

Event Message: Scaled down from 3 to 2 instances. All metrics are currently below minimum thresholds.

Event Time: 2019-04-16 13:40:04 +0000 UTC

The autoscaler will continue to scale this app within the current instance limits:

Minimum Instances: 2
Maximum Instances: 3

6. Monitoring and troubleshooting your app

- At PWS console, go to your app and click “View in PCF Metrics”.

Home / tchua / ws-test / employee-api

APP

employee-api

● Running

VIEW APP

Overview Services (2) Route (1) Logs Tasks Trace Threads Settings Buildpack: N/A

Events Last Push: 09:40 PM 04/16/19 **App Summary**

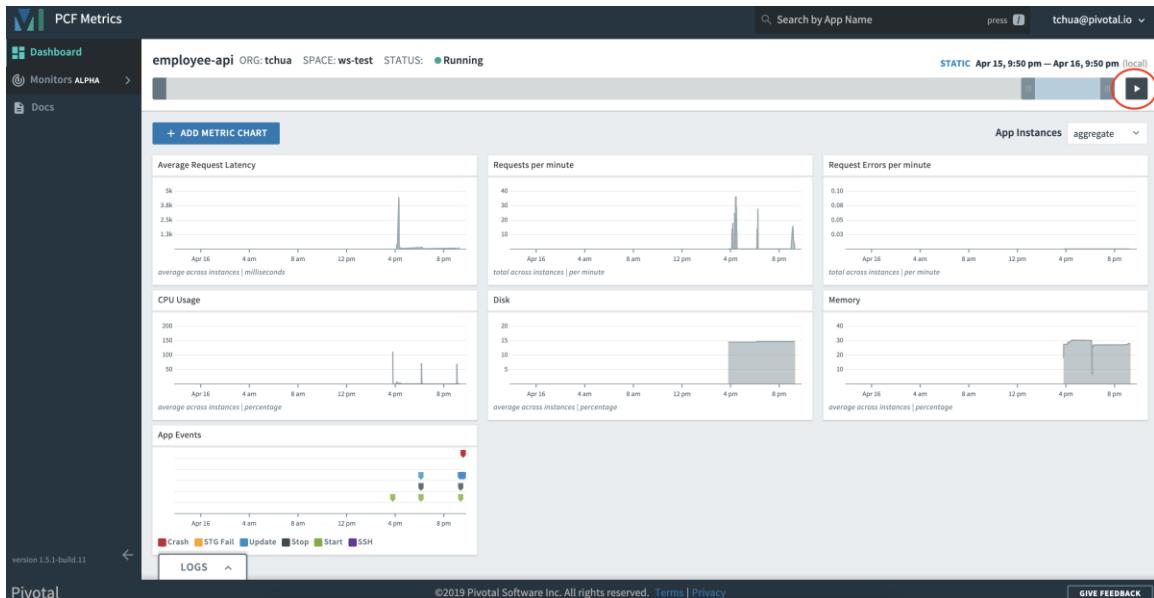
Updated app 04/16/2019 at 09:40:04 PM	Instances / Allocated 2 / 2	Memory / Allocated 0.60 / 2.00 GB	Disk / Allocated 0.29 / 2.00 GB
--	--------------------------------	--------------------------------------	------------------------------------

Processes and Instances

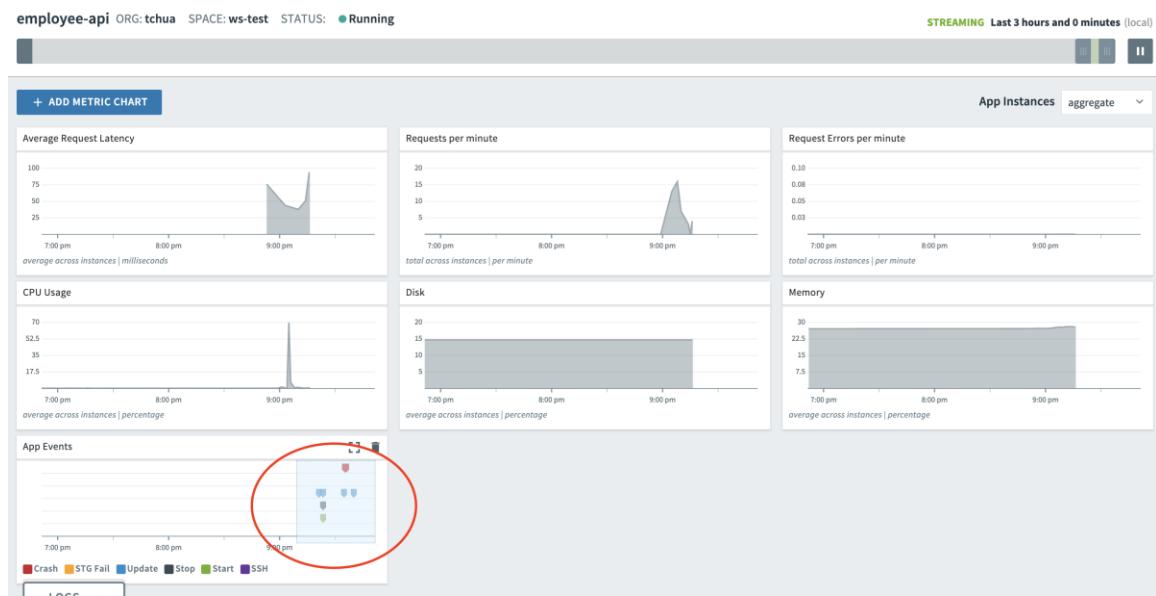
web

Instances	Memory Allocated	Disk Allocated	SCALE		
2	1 GB	1 GB			
#	App Health	CPU	Memory	Disk	Uptime
> 0		0%	308.74 MB	150.41 MB	23 min
> 1		0%	302.4 MB	150.41 MB	24 min

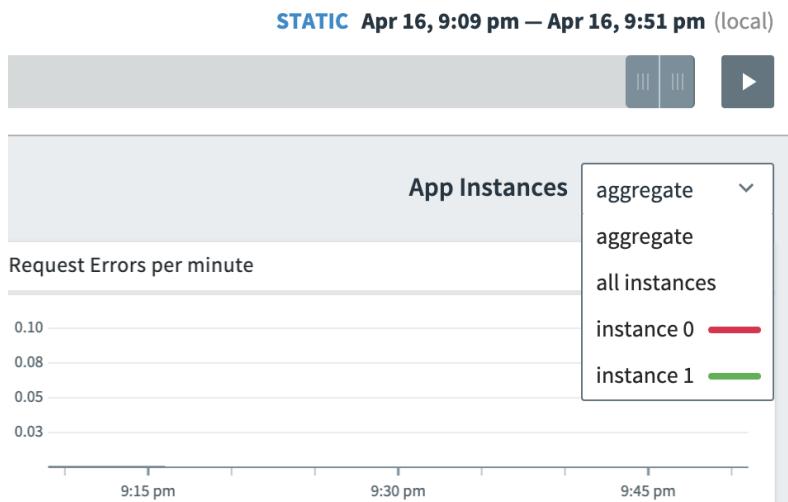
- A new tab will open up to the PCF Metrics page, showing all the different metrics that are captured per application instance. You can view the metrics at different time periods by sliding and stretching the time bar at the top. You can also click on the play button to the right of the time bar as illustrated. This will put the page into live mode, showing constantly the different performance areas of the app during the last 3 hours.



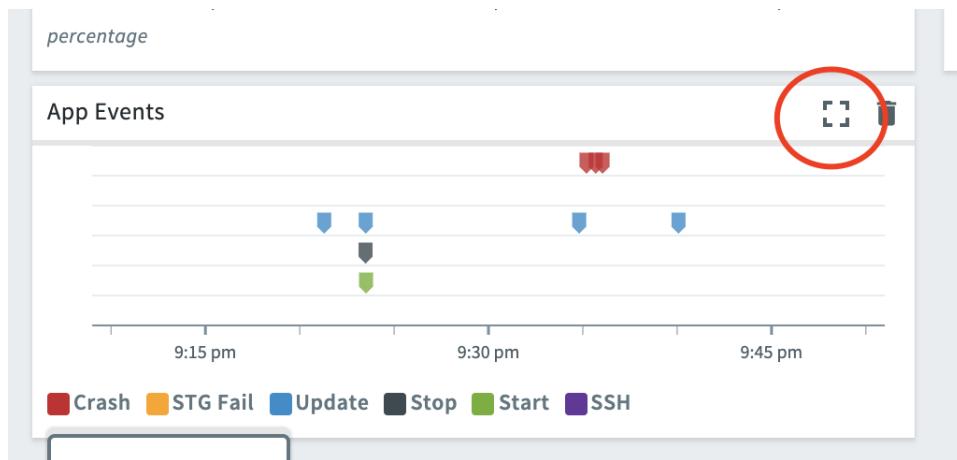
3. Instead of using the time bar, you can narrow down your focus by clicking and dragging to zoom into a specific time period as illustrated. Doing so will change the time period shown across all metrics chart.



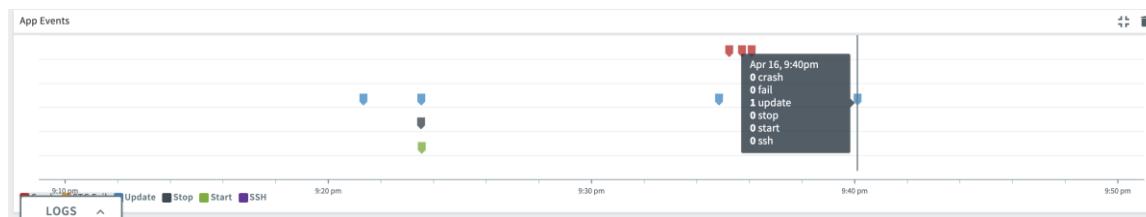
4. You can also select different views by selecting the “App Instances” drop down list as shown.



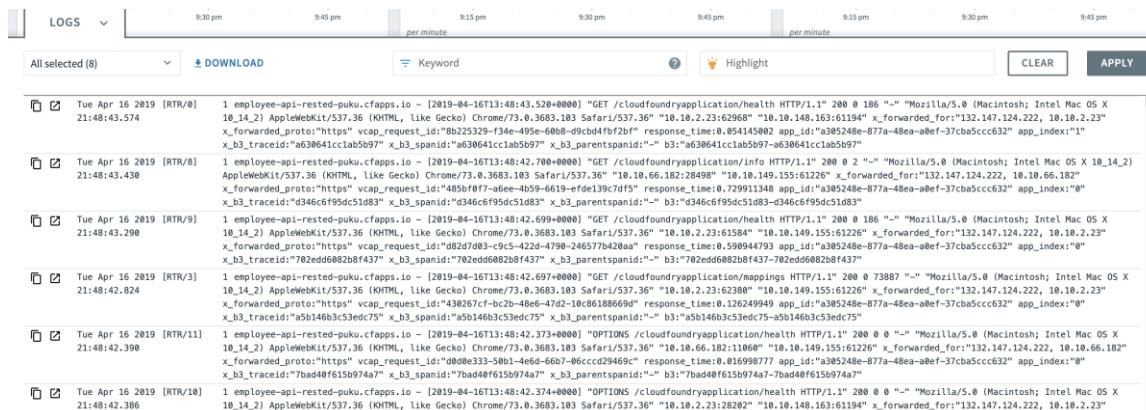
5. You can expand a chart to have more display area.



6. You can also mouse over to display relevant details pertaining to a specific metric.



7. You can also view the logs across all instances by clicking on the “Logs” tab at the bottom.



8. Within the “Logs” panel, you can do different filters, such as filtering by log types as shown. This can help filter out the noise to aid your troubleshooting.

The screenshot shows the 'LOGS' panel in the Pivotal Cloud Foundry interface. At the top, there's a dropdown menu set to 'APP' and a 'DOWNLOAD' button. Below this is a list of log sources:

- All selected (8)
- APP (Application)
- API (Cloud Controller)
- RTR (Router)
- CELL (Diego Cell)
- HEALTH (Health Check)

On the right, log entries are displayed in a table format:

		LOG
/0]		1 employee-api-rested-10_14_2) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.169 Safari/537.36
/1]		x_forwarded_proto:"http://127.0.0.1:4000"
/2]		x_b3_traceid:"a630641c5f3e4a3d83a3a3a3a3a3a3a3"
/3]		1 employee-api-rested-10_14_2) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.169 Safari/537.36
/4]		x_forwarded_proto:"http://127.0.0.1:4000"
/5]		x_b3_traceid:"d346c6f95f3e4a3d83a3a3a3a3a3a3a3"
/6]		1 employee-api-rested-10_14_2) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.169 Safari/537.36
/7]		x_forwarded_proto:"http://127.0.0.1:4000"
/8]		x_b3_traceid:"702edd605f3e4a3d83a3a3a3a3a3a3a3"
/9]		1 employee-api-rested-10_14_2) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.169 Safari/537.36
/10]		x_forwarded_proto:"http://127.0.0.1:4000"
/11]		x_b3_traceid:"a5b146b35f3e4a3d83a3a3a3a3a3a3a3"

At the bottom left, there are checkboxes for 'All' and 'Logs' with the text 'Tue Apr 16 2019 [RTR/11] 21:48:42.390'.

- You also have the option to retrieve or tail your app logs using 'cf logs' command in the cf cli. Note that the logs are automatically aggregated across all apps.

```
Derrick-Chuas-MacBook-Pro:spring-to-pcf tmchua$ cf help logs
NAME:
  logs - Tail or show recent logs for an app

USAGE:
  cf logs APP_NAME

OPTIONS:
  --recent      Dump recent logs instead of tailing

SEE ALSO:
  app, apps, ssh
Derrick-Chuas-MacBook-Pro:spring-to-pcf tmchua$
```

- Another handy feature within the logs panel are the links to the trace explorer. Clicking on a link opens up the transaction trace for that request. If a call was to traverse multiple hops across different microservices, the traversal path will show up, giving breakdown such as latency per hop. Such information is useful for understanding the cause of slowness or errors across a multi-hop call.

LOGS  150
75

All selected (8)  Keyword

View in Trace Explorer 

<input type="checkbox"/> <input checked="" type="checkbox"/> Tue Apr 16 2019 [RTR/0] 21:48:43.574	1 employee-api-rested-puku.cfapps.io - [2019-04-16T13:48:43.520+0000] "GET /cloud 10_14_2" AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.103 Safari/537.3 x_forwarded_proto:"https" vcap_request_id:"8b225329-f34e-495e-60b8-d9cbd4fbf2bf" x_b3_traceid:"a630641cc1ab5b97" x_b3_spanid:"a630641cc1ab5b97" x_b3_parentspanid:
<input type="checkbox"/> <input checked="" type="checkbox"/> Tue Apr 16 2019 [RTR/8] 21:48:43.430	1 employee-api-rested-puku.cfapps.io - [2019-04-16T13:48:42.700+0000] "GET /cloud AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.103 Safari/537.36" "10.10 x_forwarded_proto:"https" vcap_request_id:"485bf0f7-a6ee-4b59-6619-efde139c7df5" x_b3_traceid:"d346c6f95dc51d83" x_b3_spanid:"d346c6f95dc51d83" x_b3_parentspanid:
<input type="checkbox"/> <input checked="" type="checkbox"/> Tue Apr 16 2019 [RTR/9] 21:48:43.290	1 employee-api-rested-puku.cfapps.io - [2019-04-16T13:48:42.699+0000] "GET /cloud 10_14_2" AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.103 Safari/537.3 x_forwarded_proto:"https" vcap_request_id:"d82d7d03-c9c5-422d-4790-246577b420aa" x_b3_traceid:"702edd6082b8f437" x_b3_spanid:"702edd6082b8f437" x_b3_parentspanid:
<input type="checkbox"/> <input checked="" type="checkbox"/> Tue Apr 16 2019 [RTR/3] 21:48:42.824	1 employee-api-rested-puku.cfapps.io - [2019-04-16T13:48:42.697+0000] "GET /cloud 10_14_2" AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.103 Safari/537.3 x_forwarded_proto:"https" vcap_request_id:"430267cf-bc2b-48e6-47d2-10c86188669d" x_b3_traceid:"a5b146b3c53edc75" x_b3_spanid:"a5b146b3c53edc75" x_b3_parentspanid:
<input type="checkbox"/> <input checked="" type="checkbox"/> Tue Apr 16 2019 [RTR/11] 21:48:42.390	1 employee-api-rested-puku.cfapps.io - [2019-04-16T13:48:42.373+0000] "OPTIONS /c 10_14_2" AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.103 Safari/537.3 x_forwarded_proto:"https" vcap_request_id:"d0d0e333-50b1-4e6d-66b7-06cccd29469c" x_b3_traceid:"7bad40f615b974a7" x_b3_spanid:"7bad40f615b974a7" x_b3_parentspanid:

PCF Metrics

traceId=a630641cc1ab5b97

Search by App Name: tchua@pivotal.io (local)

Apr 16, 21:48:43.574 pm - Apr 16, 21:48:43.574 pm (local)

Sort by: start time

Response Time milliseconds



employee-api /cloudfoundryapplication/health 10.8 21.7 32.5 43.3 54.1

LOGS  54.1



Tue Apr 16 2019 [RTR/8] 21:48:43.574	APP employee-api 1 employee-api-rested-puku.cfapps.io - [2019-04-16T13:48:43.520+0000] "GET /cloudfoundryapplication/health HTTP/1.1" 200 8 186 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_2) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.103 Safari/537.36" "10.10.2.23:62968" "10.10.148.163:61194" x_forwarded_for:"132.147.124.222, 10.10.2.23" x_forwarded_proto:"https" vcap_request_id:"8b225329-f34e-495e-60b8-d9cbd4fbf2bf" response_time:0.054145082 app_id:"a385248e-877a-48ea-a0ef-37cba5ccc832" app_index:"1" x_b3_traceid:"a630641cc1ab5b97" x_b3_spanid:"a630641cc1ab5b97" x_b3_parentspanid:""- b3:a630641cc1ab5b97-a630641cc1ab5b97"
---	---