

Number to Word



DECEMBER 10 2019

Console, REST-API endpoint, highly scalable
Developed by: Selva Vadivel
vselvakumaranv@hotmail.com

Documentation

Quick Demo	3
Specifications	5
Download and Run	8
Testing and Quality	10
Screenshots	11
Common Troubleshooting Issues	22
Selenium Test Results	25

QUICK DEMO

1. OPTION 1: View live demo at AWS:

a. REST API using the end-point in a tool like Post-man for GET request:

`http://api.numtoward.dice.red:8080/numtoward/123`

SAMPLE OUTPUT:

```
{
  "numberInputStr": "123",
  "words": "One hundred and twenty three",
  "instanceName": "instanceAPIapp01: ID:i-01482c8d8324291a3"
}
```

b. Web-app Console URL:

<http://console.numtoward.dice.red/numtoward-mvc-dash/>

SAMPLE OUTPUT:

After clicking “Convert” button, it will take you to the results page. When you right click and view source of the page, you can see the EC2 instance ID for the Console and the instance ID for the REST-API producer. You can test it for auto-scaling and traffic from the US and non-US – it will go to different instances so to meet one of the requirements. It is configured at AWS using Route 53 – geolocation policy setting and routed to corresponding load balancers with non-scaling and auto-scaling groups behind it with EC2 instances. It is best practice to set both the US and non-US users routing load balancer to point to auto-scaling with different min and max instances for scaling instead of making one load balancer point to non-scaling set of instance and the other to auto scaling group. While it is using only http here, it is best practice to order a SSL certificate and use https for having an encrypted communication.

COMMON TROUBLE SHOOTING:

You may get a 404 error. Please try again using

<http://console.numtoward.dice.red/numtoward-mvc-dash/> for the console or the

REST API endpoint. This can be built in as a feature to try twice on a 404 error but it is not included in this present release.

2. OPTION 2: Setup demo on your own:

- a. Check out code from repo

<https://repo.dice.red/vselvakumaranv/sonatype>

- b. Run Maven build at numtowards-master. This will create the jars for numtowards-api and numtoward-mvc-dash in their respective target directories.
- c. Copy the generated JAR file for API and WAR file for the console to the locations where you want them to run.
- d. In the same folder of each of the jar file, create a folder called “config”. Inside the config folder create a file called application.properties.
- e. For the application.properties file of numtowards-api use the following. You may change these ports and name as you deem appropriate for each running instance:

```
server.port=8081
spring.application.name=NumToWordAPIproduces
appInstance.name=instanceAPIapp01
```

- f. Run the command `java -jar {numtowards-api-filename}.jar`. Now the application end point can be accessed from a tool like Post-man using the following URL:

<http://localhost:8081/numtoward/11231>

- g. For the application.properties file inside the WAR file of numtoward-mvc-dash, update the api.hostname variable to the above endpoint.

SPECIFICATIONS

I DESCRIPTION

This tool converts numbers into its corresponding words. This tool can be used using

1. The REST-API GET endpoint or using the console online.
2. The console consumes the REST-API endpoint.

II REQUIREMENTS

Commas and hyphens (indicating negative input) are the only non-numeric characters that are considered valid input. With commas, there are extra rules, however. For example, 1,000 is valid input, but 10,00 would be invalid. Only need to support whole numbers. Negative numbers are valid input as well. However, fractional, irrational, and decimal numbers can be treated as invalid input. Only need to support values that fit within a signed 32-bit integer. For java, then the range would be from `java.lang.Integer.MIN_VALUE` to `java.lang.Integer.MAX_VALUE` (inclusive). There is no specific set of frequent numbers. Reviewers will try various combinations to try to crash your submission and to make sure that all values are properly supported. An API alone isn't quite enough, would like some means included that lets a user execute the program. A single number at a time is what is expected as input at a time, other forms of input is invalid. An end-user will not expect to use the program without an internet connection. There is no authorization requirement or rate limiting. There will be no analytics attached to the program. Users will typically only submit a single request at a time. The peak time for requests is during school hours in the US - in particular, during the hours when all schools are in session (roughly 11am to 3pm eastern time). The only future direction is that for the program to be eventually included in popular, free Linux distributions such as Debian, Ubuntu, and Fedora.

III ANALYSIS OF REQUIREMENTS

To meet above requirement, the program was developed in Java using Spring framework. It was deployed in AWS with Auto-scaling for US users and setting Route 53 with policy for routing users by the geolocation to non-scaling EC2 instances and

auto-scaling groups behind load balancers from 11 am to 3 pm EST, which is 16:00 - 20:00 UTC. There is really no need for logging as part of the Java application. Also, Spring Security dependency is not required as there is no authorization and authentication requirements. AWS Inspector to test vulnerabilities for both the network and application is only needed in addition to Application level testing using Junit, Selenium and input pattern validation.

IV CODE

The code repository of this tool is at Gitlab. You may download the tool and open it in an IDE like Eclipse. The tool was developed using Spring Tool Suite 3 for Eclipse. The tool has been setup as a multi-module Maven project. The 'numtowards-master' project has five modules under it. They are:

1. numtowards-service-word
2. numtowards-api
3. numtowards-domain

'numtowards-api' produces the REST-API GET endpoint. Please look into its application.properties file for configuration settings. 'numtowards-api' has a dependency on 'numtowards-service-word'. 'numtowards-service-word' provides the core functionality to convert number to words. It has optimized number of JUnit test cases to validate the boundary conditions for both valid and invalid inputs.

'numtowards-mvn-dash' consumes the REST-API GET provided by 'numtowards-api'. It has numberConsole.jsp which will be used by end users to enter the numbers. Then this tool validates user input for the format. The text box is already limited to 14 characters which is the maximum character size of the input as per the set boundary conditions and provides validation with feedback to enhance user experience. The api and console are separated for both software functionality and hardware resource so to support loosely coupled architecture.

“Logging: For this application, AWS CloudWatch is sufficient as no additional information is needed, so additional logging is not part of the application”

DOWNLOAD AND RUN

To get started right away with setting up a demo on own machine, follow the below instructions.

Step 1: Meeting the following system requirements:

- a. JDK 8
- b. Spring Tool Suite 3 - Eclipse IDE
- c. Dependencies are added in pom.xml. In particular:
 - i. AWS SDK for Java 1.11 is included as a dependency.
 - ii. Undertow is included as dependency in the pom.xml for API and the default Tomcat is excluded.
- d. Apache Maven 3.6.3

Step 2: The code can be downloaded from the GitLab repo.

Step 3: Open in an IDE like Eclipse. It is recommended to use Spring Tool Suite 3 for Eclipse as the tool was built using it. Java 8 JDK, Apache Tomcat 9.0.29-windows-x64 and Apache Maven 3.6.3 were used for developing this tool. Tomcat was later excluded and Undertow was included as dependency in pom.xml. This was done to improve the performance of the tool and extend the support for Event Driven Architecture.

Step 4: If you are going to test locally, first run the 'numtowards-api' as Springboot app. Then run 'numtoward-mvc-dash' as a Java Application. There is one application.properties file for each 'numtowards-api' and 'numtoward-mvc-dash'. These application.property files have the settings for these two modules. It may be noted that the configuration is set to use different port numbers : 8081 and 8082 for these two modules and not the default port 8080 for Tomcat.

Step 4: Set the goal to 'clean install' in case you have not already set it for any of the modules. Maven build 'numtowards-api' and it will create a runnable jar that can be deployed where you would like to run it. Select Maven clean if there is any error which is common because of IDE/Eclipse cache settings. Similar it can be done for 'numtoward-mvc-dash' as well.

Step 5: In the result page of the console, it will also display the system information. The system information will include the console instance name, console EC2 instance's ID, API

instance name and API EC2 instance's ID. In a load balanced setup, these values may change for each request.

Note:

For one of the requirements where the usage is expected to be higher from 11:00 am to 3:00 pm EST in the US, the auto-scaling group is set for that time period at AWS live demo setup.

Other Details:

1. AWS SDK for Java 1.11 is needed to get the instance ID of nodes that are processing the API and Console. It is used to only validate one of the requirements where more users from the US are expected to use from 11:00 am – 3:00 pm EST. So Auto-scaling group was set at AWS behind a load-balancer, and also set other traffic to another non-scaling load balancer based on geo-location policy of Route 53. The instance ID of those non-scaling and auto-scaling are returned for you to validate that this requirement is met to handle growing users from the US during business hours listed above. But including AWS DSK as dependency as it is set now will cause the size of build of WAR and JAR files to be larger than your expectation. As an alternative, you may install this kit locally if you prefer.
2. If you would like to use the default Tomcat instead of Undertow, you may remove the <exclusion> tag for Tomcat and remove dependency tag for Undertow from pom.xml to default to Tomcat. This application was tested using Apache Tomcat 9.0.29-windows-x64 as well to compare for performance.

TESTING AND QUALITY

The following are the different ways by which this tool was tested, and high-quality tool was developed. The following modules are opened in Eclipse IDE to run the tests.

I. REST-API ENDPOINT TESTING

1. JUnit at 'numtowards-service-word' module performs 13 different test cases. It will be run every time Maven build is run.

II. APPLICATION CONSOLE TESTING

2. Selenium webdriver for Google Chrome browser is setup in the module 'numtowards-console-auto-test'. In eclipse/IDE select that module and run it as a Java Application. You can see the demo of the testing where the tool will open the browser, go through the events and test for the different cases written as methods.
3. Validation at numtoward-mvc-dash module for the input text field. @Pattern . annotation and other special character checks as performed. Appropriate error message is displayed on the webpage if the pattern is not matched. Also, the textbox is limited to 14 characters to meet the boundary conditions.

Test Results of Selenium are provided in page 25.

SCREENSHOTS

Descriptions are added in the below screenshots when needed, otherwise it should be self-explanatory.

In case you prefer using the Maven CLI, then just use the regular mvn commands to clean and run the build. If you prefer using the IDE, see below.

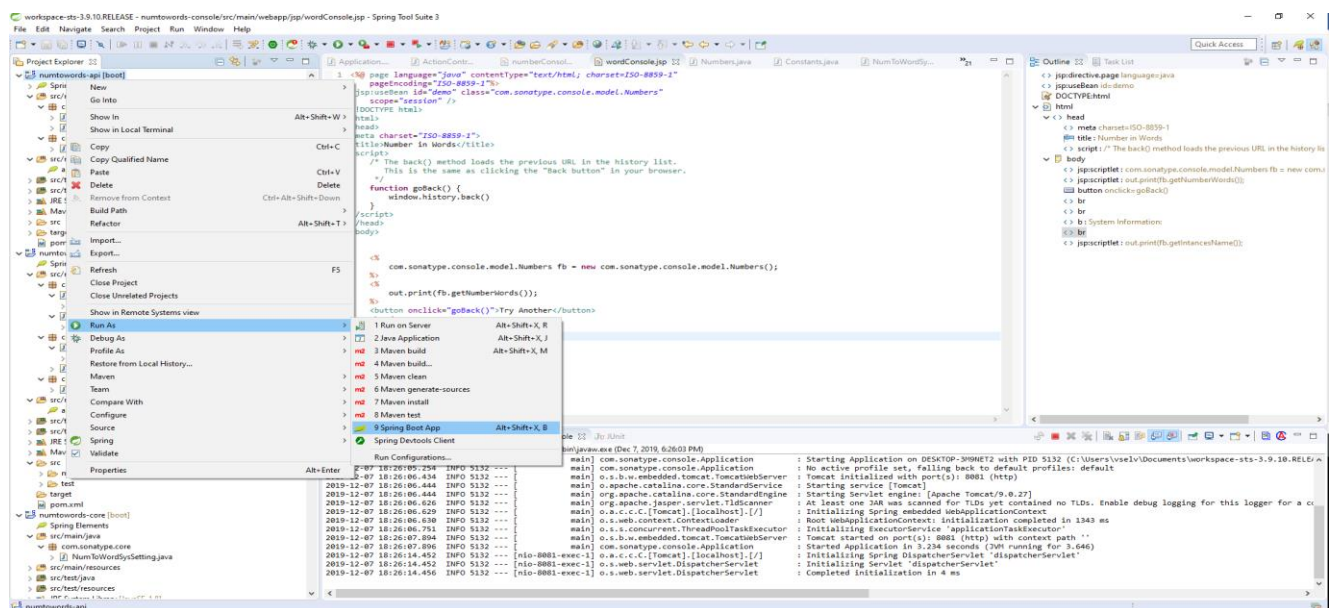


Figure 1: The above screenshot shows how to start the API from Eclipse IDE with Spring Tool Suite 3

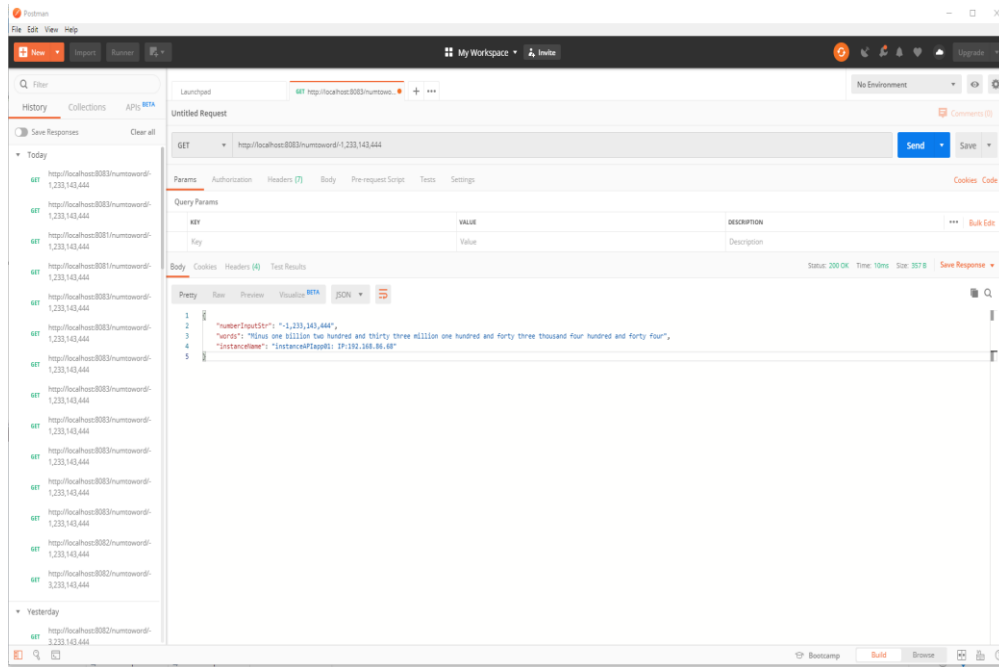


Figure 2: The above screenshot shows the response for REST-API endpoint at POSTMAN

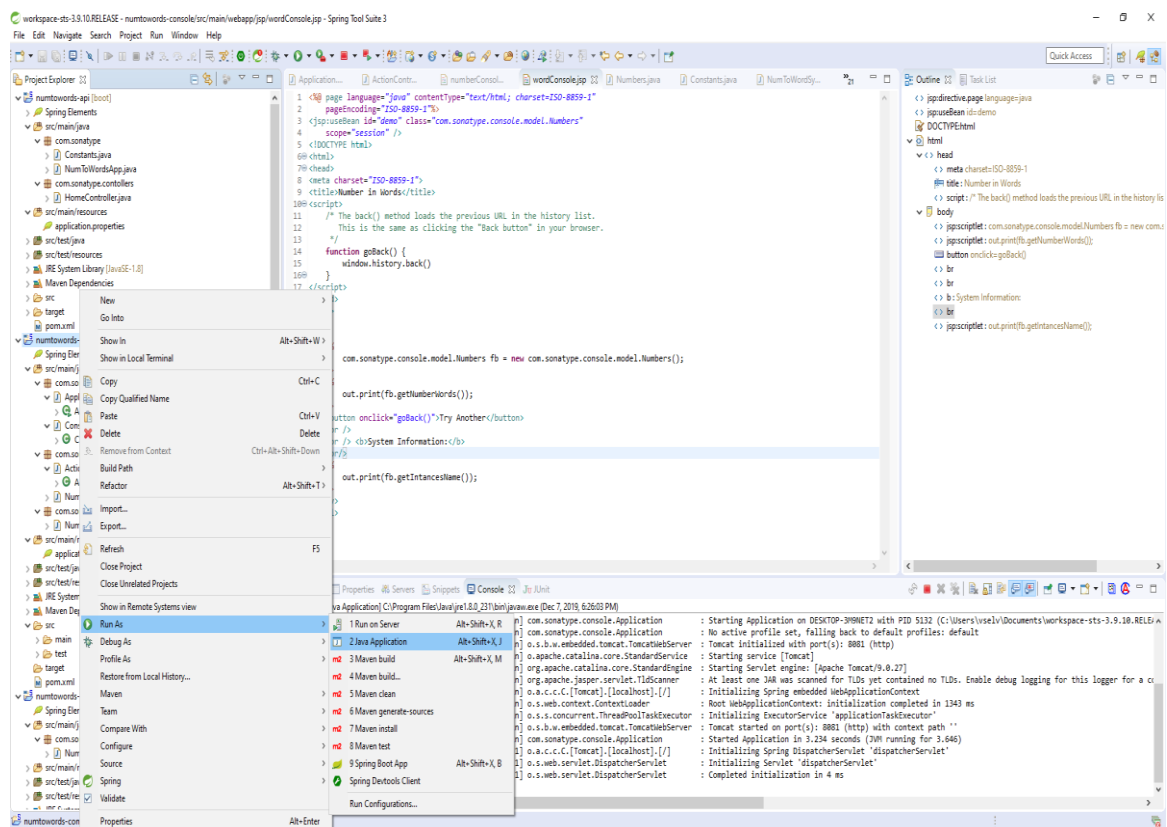


Figure 3: The above screenshot shows how to start the console from Eclipse IDE. You may also export it as a WAR file, or do a Maven build.

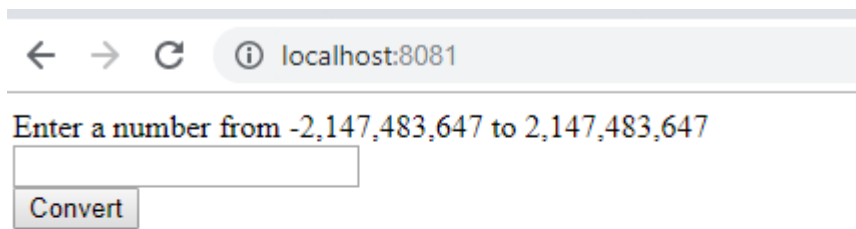


Figure 4: The above screenshot shows console at Google Chrome browser.

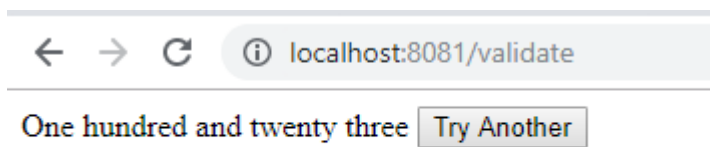
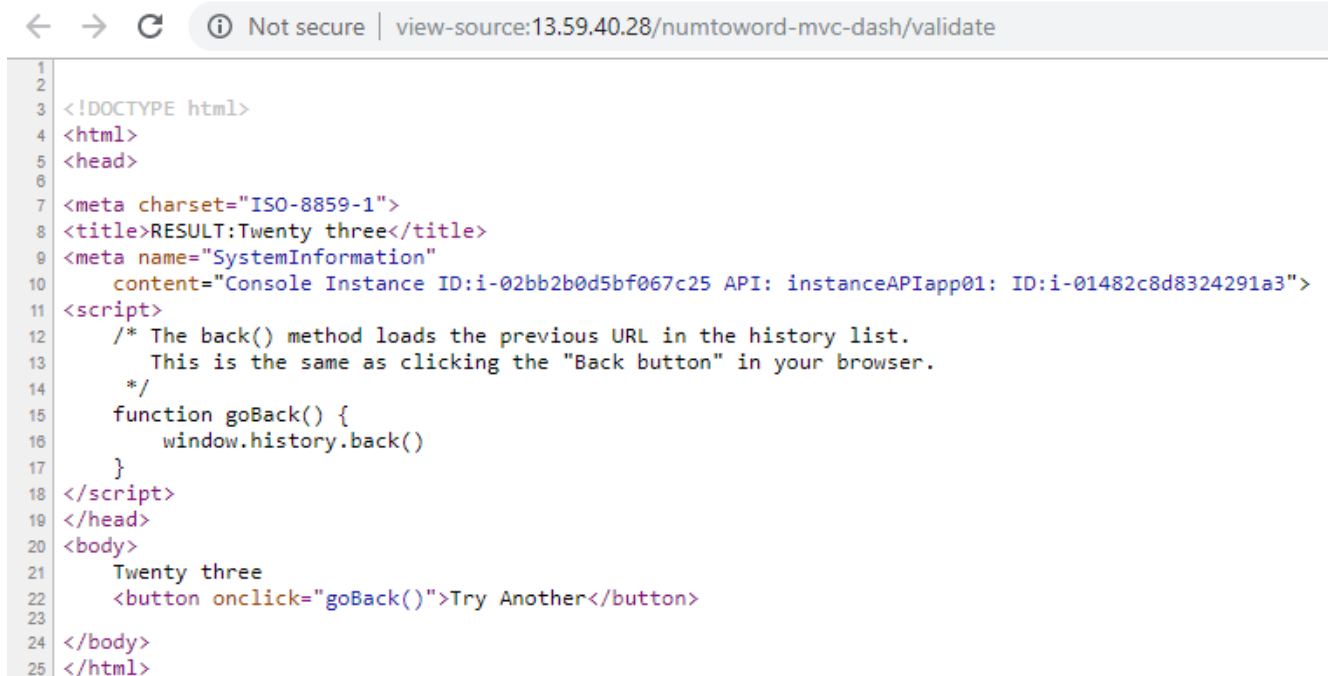


Figure 5: The above screenshot shows console result page at Google Chrome browser.




```
1
2
3 <!DOCTYPE html>
4 <html>
5 <head>
6
7 <meta charset="ISO-8859-1">
8 <title>RESULT:Twenty three</title>
9 <meta name="SystemInformation"
10   content="Console Instance ID:i-02bb2b0d5bf067c25 API: instanceAPIapp01: ID:i-01482c8d8324291a3">
11 <script>
12   /* The back() method loads the previous URL in the history list.
13      This is the same as clicking the "Back button" in your browser.
14   */
15   function goBack() {
16     window.history.back()
17   }
18 </script>
19 </head>
20 <body>
21   Twenty three
22   <button onclick="goBack()">Try Another</button>
23
24 </body>
25 </html>
```

Figure 6: The above screenshot shows source code in the browser for the console result page at Google Chrome browser. The information about the instance Console and instance for API can be obtained here at the meta tag. This can be used to validate the working of load balancers with AWS EC2 instances. The console and API are loosely coupled so that there can be load balancers for each of them separately and improve end user experience with very quick response. In the above example the instance ID i-02bb2b0d5bf067c25 is that of the console and the instance ID i-01482c8d8324291a3 is the instance of the API producer. Even though the load balancer will auto scale for US users and not in the case of other non-US users, the instances between non-scaling and auto-scaling can be shared between the two users in the US and non-US.

Auto Scaling Group Setup at AWS:

There is no information on predicted amount of traffic. So, the number of instances for maximum is selected to a low number and alarm is set so that this can be updated once the utilization has increased and it is within budget to increase the number of instances.


 Services ▾ Resource Groups ▾ ⌕

1. Configure Auto Scaling group details 2. Configure scaling policies 3. Configure Notifications 4. Configure Tags 5. Review

Create Auto Scaling Group

Launch Configuration ⓘ AutoScaleAPI

Group size ⓘ Start with instances

Network ⓘ  [Create new VPC](#)

Subnet ⓘ

[Create new subnet](#)

Each instance in this Auto Scaling group will be assigned a public IP address. ⓘ

▼ Advanced Details

Load Balancing ⓘ ☒ Receive traffic from one or more load balancers [Learn about Elastic Load Balancing](#)

Classic Load Balancers ⓘ


Target Groups ⓘ

Health Check Type ⓘ ☒ ELB ☐ EC2

Health Check Grace Period ⓘ seconds

Monitoring ⓘ Amazon EC2 Detailed Monitoring metrics, which are provided at 1 minute frequency, are not enabled for the launch configuration AutoScaleAPI. Instances launched from it will use Basic Monitoring metrics, provided at 5 minute frequency.
[Learn more](#)

Instance Protection ⓘ

Service-Linked Role ⓘ  [View Role in IAM](#)

Create Auto Scaling group

Actions ▾

Filter:

<input type="checkbox"/>	Name	Launch Configuration /	Instances	Desired	Min	Max	Availability Zones	Default Cooldown	Health Check Grace
<input type="checkbox"/>	APIAutoScale...	api-reboot-auto-config-...	2	2	1	3	us-east-2a, us-east-2b	300	300
<input checked="" type="checkbox"/>	autoScaleUS	AutoScaleAPI	2	2	1	2	us-east-2a, us-east-2b	300	300

Add policy

Decrease Group Size

Policy type: Step scaling

Execute policy when: awsec2-autoScaleUS-High-CPU-Utilization
breaches the alarm threshold: CPUUtilization <= 50 for 300 seconds
for the metric dimensions AutoScalingGroupName = autoScaleUS

Take the action: Remove 1 capacity units when >= CPUUtilization > -infinity

Increase Group Size

Policy type: Step scaling

Execute policy when: awsec2-autoScaleUS-CPU-Utilization
breaches the alarm threshold: CPUUtilization >= 80 for 300 seconds
for the metric dimensions AutoScalingGroupName = autoScaleUS

Take the action: Add 1 capacity units when 80 <= CPUUtilization < +infinity

Instances need: 300 seconds to warm up after each step

Services ▾ Resource Groups ▾

Try the new design for Amazon EC2 Auto Scaling

This older console is being replaced with the new EC2 Auto Scaling console. No new features or improvements will be made in this older console. [Go to the new console.](#)

Create Auto Scaling group

Actions ▾

Filter:

<input type="checkbox"/>	Name	Launch Configuration /	Instances	Desired	Min	Max	Availability Zones	Default Cooldown	Health Check Grace
<input type="checkbox"/>	APIAutoScale...	api-reboot-auto-config-...	2	2	1	3	us-east-2a, us-east-2b	300	300
<input checked="" type="checkbox"/>	autoScaleUS	AutoScaleAPI	2	2	1	2	us-east-2a, us-east-2b	300	300

Add policy

Decrease Group Size

Policy type: Step scaling

Execute policy when: awsec2-autoScaleUS-High-CPU-Utilization
breaches the alarm threshold: CPUUtilization <= 50 for 300 seconds
for the metric dimensions AutoScalingGroupName = autoScaleUS

Take the action: Remove 1 capacity units when >= CPUUtilization > -infinity

Increase Group Size

Policy type: Step scaling

Execute policy when: awsec2-autoScaleUS-CPU-Utilization
breaches the alarm threshold: CPUUtilization >= 80 for 300 seconds
for the metric dimensions AutoScalingGroupName = autoScaleUS

Take the action: Add 1 capacity units when 80 <= CPUUtilization < +infinity

Instances need: 300 seconds to warm up after each step

<div> <div>Create Auto Scaling group</div> <div>Actions</div> </div>									
<div> <div>Filter: <input type="text" value="Filter Auto Scaling groups..."/></div> <div> <div><<</div> <div><</div> <div>1 to 2 of 2 Auto Scaling Groups</div> <div>></div> <div>>></div> </div> </div>									
<input type="checkbox"/>	Name	Launch Configuration / -	Instances	Desired	Min	Max	Availability Zones	Default Cooldown	Health Check Grace
<input type="checkbox"/>	APIAutoScale...	api-reboot-auto-config...	2	2	1	3	us-east-2a, us-east-2b	300	300
<input checked="" type="checkbox"/>	autoScaleUS	AutoScaleAPI	2	2	1	2	us-east-2a, us-east-2b	300	300

Auto Scaling Group: autoScaleUS

Details

Activity History

Scaling Policies

Instances

Monitoring

Notifications

Tags

Scheduled Actions

Lifecycle Hooks

Create Scheduled Action

Actions

<div> <div>Filter: <input type="text" value="Filter scheduled actions..."/></div> <div> <div><<</div> <div><</div> <div>1 to 2 of 2 Scheduled Actions</div> <div>></div> <div>>></div> </div> </div>									
<input type="checkbox"/>	Name	Start Time	End Time	Recurrence	Desired Capacity	Min	Max		
<input checked="" type="checkbox"/>	autoScaleUSTimeline	2019 December 11 11:00:00 UTC-5		0 16 * * *	2	1	3		
<input type="checkbox"/>	OffAutoScaleUS	2019 December 11 15:00:00 UTC-5		0 20 * * *	1	1	1		

The above figure shows that the auto-scaling of the cron is set between 11:00 am and 3:00 pm EST (i.e. 16:00-20:00 UTC) to meet the requirements.

aws

Services

Resource Groups

1. Configure Load Balancer

2. Configure Security Settings

3. Configure Security Groups

4. Configure Routing

5. Register Targets

6. Review

Step 1: Configure Load Balancer

Basic Configuration

To configure your load balancer, provide a name, select a scheme, specify one or more listeners, and select a network. The default configuration is an Internet-facing load balancer in the selected network with a listener that receives HTTP traffic on port 80.

Name

AutoScaleELBConsole

Scheme

☒ internet-facing
 ☐ internal

IP address type

ipv4

Listeners

A listener is a process that checks for connection requests, using the protocol and port that you configured.

Load Balancer Protocol	Load Balancer Port
HTTP	80
HTTPS (Secure HTTP)	443

Add listener

Availability Zones

Specify the Availability Zones to enable for your load balancer. The load balancer routes traffic to the targets in these Availability Zones only. You can specify only one subnet per Availability Zone. You must specify subnets from at least two Availability Zones to

VPC

vpc-96f10efd (172.31.0.0/16) (default)

Availability Zones

☒ us-east-2a

subnet-f0cc2b9b

IPv4 address

Assigned by AWS

☒ us-east-2b

subnet-c181dcbb

IPv4 address

Assigned by AWS

☐ us-east-2c

subnet-5632b71a

IPv4 address

Assigned by AWS

Tags

Configuring Load Balancer

aws

Services

Resource Groups

1. Configure Load Balancer

2. Configure Security Settings

3. Configure Security Groups

4. Configure Routing

5. Register Targets

6. Review

Step 2: Configure Security Settings

Select default certificate

AWS Certificate Manager (ACM) is the preferred tool to provision and store server certificates. If you previously stored a server certificate using IAM, you can deploy it to your load balancer. [Learn more](#) about HTTPS listeners and certificate management.

Certificate type

☒ Choose a certificate from ACM (recommended)

☐ Upload a certificate to ACM (recommended)

☐ Choose a certificate from IAM

☐ Upload a certificate to IAM

Request a new certificate from ACM

AWS Certificate Manager makes it easy to provision, manage, deploy, and renew SSL Certificates on the AWS platform. ACM manages certificate renewals for you. [Learn more](#)

Certificate name

quality.dice.red (arn:aws:acm:us-east-2:192280676785:certificate/010e6)

Select Security Policy

Security policy

ELBSecurityPolicy-2016-08

Configuring Load Balancer

Step 5: Register Targets

Register targets with your target group. If you register a target in an enabled Availability Zone, the load balancer starts routing requests to the targets as soon as the registration process completes and the target passes the initial health checks.

Registered targets

To deregister instances, select one or more registered instances and then click Remove.

Remove

<input type="checkbox"/>	Instance	Name	Port	State	Security groups	Zone
No instances available.						

Instances

To register additional instances, select one or more running instances, specify a port, and then click Add. The default port is the port specified for the target group. If the instance is already registered on the specified port, you must specify a different port.

Add to registered on port 80

Q Search Instances X

<input type="checkbox"/>	Instance	Name	State	Security groups	Zone	Subnet ID	Subnet CIDR
<input type="checkbox"/>	i-09752f3595b7a7bc1	AutoScaleConsole	running	Tomcat Certified by Bltn...	us-east-2a	subnet-f0cc2b9b	172.31.0.0/20
<input type="checkbox"/>	i-02bb2bd5bf067c25	Console Non-Auto scale	running	Tomcat Certified by Bltn...	us-east-2b	subnet-c181dcb	172.31.16.0/20
<input type="checkbox"/>	i-03cbae4401464e07f	AutoScaleConsole	running	Tomcat Certified by Bltn...	us-east-2b	subnet-c181dcb	172.31.16.0/20
<input type="checkbox"/>	i-01482c8d8324291a3	API Non-Auto Scale Node	running	BigDL deep learning fra...	us-east-2c	subnet-5632b71a	172.31.32.0/20
<input type="checkbox"/>	i-0f824f0404cf7ba9d	Console US Auto Scale	running	Tomcat Certified by Bltn...	us-east-2a	subnet-f0cc2b9b	172.31.0.0/20
<input type="checkbox"/>	i-090620324b982bb95	Quality Testing Node	running	SonarQube Certified by ...	us-east-2a	subnet-f0cc2b9b	172.31.0.0/20

Configuring Load Balancer

The screenshot shows the AWS Management Console interface for configuring a Load Balancer. The left sidebar contains navigation links for various AWS services. The main content area displays a table of load balancers, with 'cELBAutoScaleConsole' selected. Below the table, the configuration details for the selected load balancer are shown, including its name, DNS name, state, VPC ID, availability zones, type, and creation time. The 'Basic Configuration' section shows the load balancer is of type 'Classic (Migrate Now)' and is internet-facing. The 'Port Configuration' section shows it is configured for HTTP on port 80.

Configuring Load Balancer

Configure actions

Notification

Whenever this alarm state is...
Define the alarm state that will trigger this action

☒ **in Alarm**
The metric or expression is outside of the defined threshold.

☐ **OK**
The metric or expression is within the defined threshold.

☐ **INSUFFICIENT_DATA**
The alarm has just started or not enough data is available.

Select an SNS topic
Define the SNS (Simple Notification Service) topic that will receive the notification

☒ **Select an existing SNS topic**

☐ Create new topic

☐ Use topic ARN

Send a notification to...

Only email lists for this account are available

Email (endpoints)
selva@dice.red - [View in SNS Console](#)

Add notification

Auto Scaling action

Whenever this alarm state is...
Define the alarm state that will trigger this action

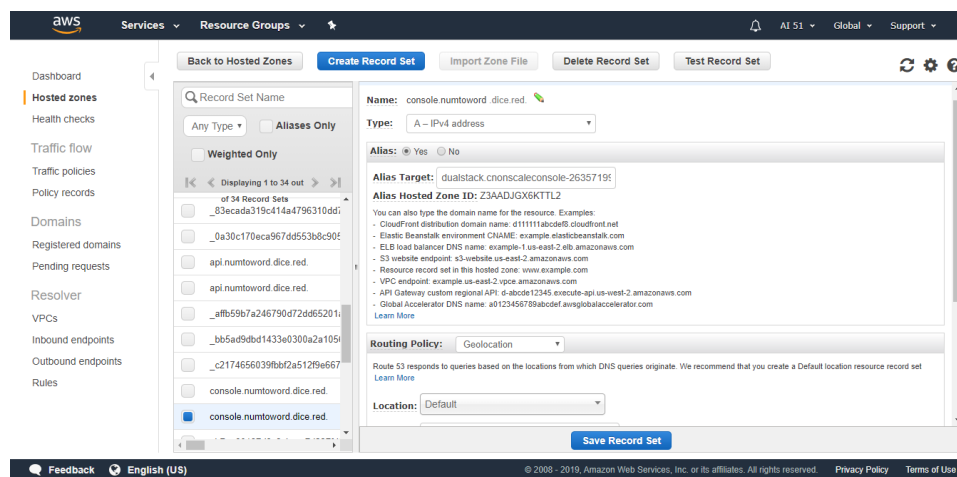
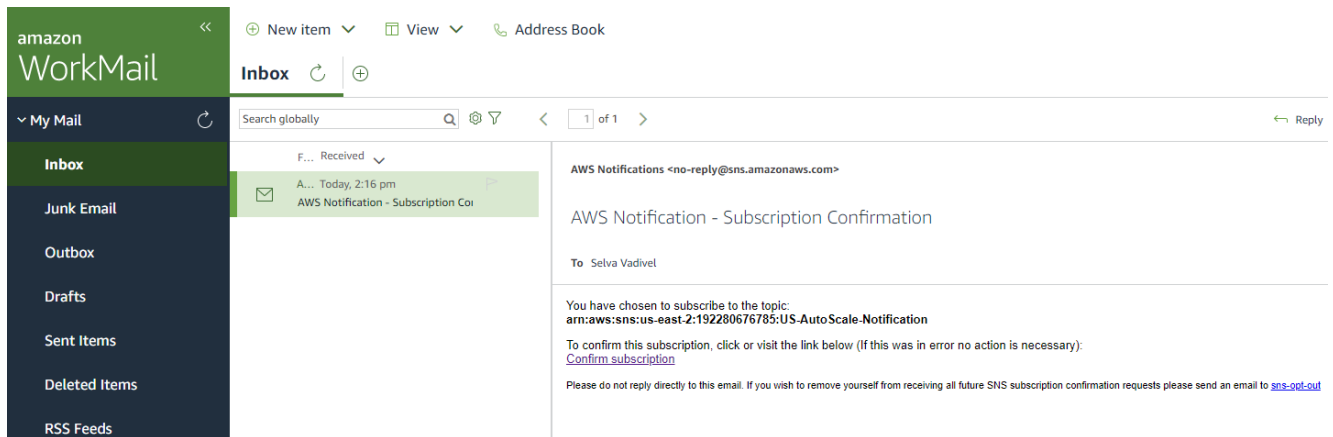
☒ **in Alarm**
The metric or expression

☐ **OK**
The metric or expression

☐ **INSUFFICIENT_DATA**
The alarm has just started

Setting Alarm at Cloud Watch

The above screenshot shows Alarm set at Cloud Watch to send email through SNS - topic and subscription is added for this Auto Scale. Confirm subscription for the confirmation email as below:



The above shows setting the URL to point to non-scaling load balancer for default users based on geolocation, it will include US users as well. It is actually best practice to auto-scale both the non-US and US users as it will not cost more.

The screenshot displays the AWS Management Console interface for creating a new record set. The left-hand navigation pane shows various AWS services, with 'Hosted zones' selected. The main content area is titled 'Create Record Set' and includes several tabs: 'Back to Hosted Zones', 'Create Record Set' (active), 'Import Zone File', 'Delete Record Set', and 'Test Record Set'. The 'Record Set Name' field is populated with 'console.numtoword.dice.red'. The 'Type' is set to 'A - IPv4 address'. The 'Alias' option is selected as 'Yes', and the 'Alias Target' is 'dualstack.elbautoscaleconsole-36477'. The 'Alias Hosted Zone ID' is 'Z3AADJGX6KTTL2'. The 'Routing Policy' is set to 'Geolocation', and the 'Location' is 'United States'. A 'Save Record Set' button is located at the bottom right of the form. The footer of the console shows the AWS logo, 'Feedback', 'English (US)', and copyright information for 2008-2019.

The above shows setting the URL to point to auto-scaling load balancer for the US users based on geolocation.

COMMON TROUBLESHOOTING ISSUES

1. Goals not set when building in IDE like Eclipse.

Solution: Set the goals to “clean install” and run Maven build.

2. No compiler is provided in the environment

Solution:

1. On your Eclipse IDE, go into Window > Preferences > Java > Installed JREs > and check your installed JREs. You should have an entry with a JDK there.
2. Select the Execution Env as show below. Click OK
3. Then Right-Click on your Project -> Maven -> Update Project

3. One or more files showing error

Solution: Run a Maven clean and Maven -> Update Project

4. The REST-API GET endpoint provides valid JSON as response for any valid number. and it returns appropriate JSON for invalid requests as well.

However, there may be cases where it will return a 404 error or no response for a very poorly constructed GET request. A REST-API endpoint is expected to be used programmatically, so further validations are expected to be provided by the program that will be consuming this end-point.

5. Missing custom jar files

Solution: Running maven build for pom.xml for numtowards-master should work in general. In case there is issue, then run maven build for pom.xml for numtowards-service-word first and then run it for numtowards-api. It is because numtowrods-api uses the jar file of service.

The console, numtowards-mvc-dash, is a separate project and is not part of the multi-module maven for api. For the console project, you can either export it as a WAR file or run maven build and that should generate the WAR file. Do not forget to update the URL in the application.properties file inside WAR file to point to the REST-API endpoint where you are running it. Also to use custom ports for the api, create a config folder in the same parent folder where the api jar is going to be run and then place the application.properties file inside the config folder, and then enter the service.port number that you would like to use. Both the api and console have application.properties file that you can customize to meet your requirements.

-
6. In case Instance ID is different from previously validated instance IDs, it is possible that the auto scaling group set a new instance. Also auto-scaling and non-scaling groups share the same instance as load balancing is based on capacity utilization and is not on round robin.
 7. When setting up the hostname for REST API application, the DNS of load balancer may resolve 404 as error because REST API end point is a variable.
Solution: Set both 200 and 404 as valid response
 8. Running Commands on Your Linux Instance at Launch:
Auto scaling may not start the jar service on reboot. So include the following command when launching a new EC2 instance or for the image of AMI for help auto-scale restart the service for API endpoint when it tries to maintain the fleet of EC2 instances using the image it has in its configuration.

```
#!/bin/bash  
cd /home/ubuntu/sonatype  
java -jar numtowards-api-0.0.1-SNAPSHOT.jar
```

numtowards-api-0.0.1-SNAPSHOT is the name of the generated jar file for the REST-API. Please update the path as per your Linux instance. The above command is not needed for a WAR file because the Tomcat will install it.

APPENDIX

SELENIUM TEST RESULTS

Starting ChromeDriver 78.0.3904.105 (60e2d8774a8151efa6a00b1f358371b1e0e07ee2-refs/branch-heads/3904@{#877}) on port 46424

Only local connections are allowed.

Please protect ports used by ChromeDriver and related test frameworks to prevent access by malicious code.

[1576037079.996][WARNING]: Timed out connecting to Chrome, retrying...

Dec 10, 2019 11:04:45 PM org.openqa.selenium.remote.ProtocolHandshake createSession

INFO: Detected dialect: W3C

Test #1 Passed!

Starting ChromeDriver 78.0.3904.105 (60e2d8774a8151efa6a00b1f358371b1e0e07ee2-refs/branch-heads/3904@{#877}) on port 34422

Only local connections are allowed.

Please protect ports used by ChromeDriver and related test frameworks to prevent access by malicious code.

[1576037101.680][WARNING]: Timed out connecting to Chrome, retrying...

Dec 10, 2019 11:05:06 PM org.openqa.selenium.remote.ProtocolHandshake createSession

INFO: Detected dialect: W3C

Test #2 Passed!

Starting ChromeDriver 78.0.3904.105 (60e2d8774a8151efa6a00b1f358371b1e0e07ee2-refs/branch-heads/3904@{#877}) on port 32683

Only local connections are allowed.

Please protect ports used by ChromeDriver and related test frameworks to prevent access by malicious code.

[1576037124.738][WARNING]: Timed out connecting to Chrome, retrying...

Dec 10, 2019 11:05:28 PM org.openqa.selenium.remote.ProtocolHandshake createSession

INFO: Detected dialect: W3C

Test #3 Passed!

Starting ChromeDriver 78.0.3904.105 (60e2d8774a8151efa6a00b1f358371b1e0e07ee2-refs/branch-heads/3904@{#877}) on port 37362

Only local connections are allowed.

Please protect ports used by ChromeDriver and related test frameworks to prevent access by malicious code.

[1576037148.272][WARNING]: Timed out connecting to Chrome, retrying...

Dec 10, 2019 11:05:50 PM org.openqa.selenium.remote.ProtocolHandshake createSession

INFO: Detected dialect: W3C

Test #4 Passed!

Starting ChromeDriver 78.0.3904.105 (60e2d8774a8151efa6a00b1f358371b1e0e07ee2-refs/branch-heads/3904@{#877}) on port 31273

Only local connections are allowed.

Please protect ports used by ChromeDriver and related test frameworks to prevent access by malicious code.

[1576037180.557][WARNING]: Timed out connecting to Chrome, retrying...

Dec 10, 2019 11:06:29 PM org.openqa.selenium.remote.ProtocolHandshake createSession

INFO: Detected dialect: W3C

Test #5 Passed!

Starting ChromeDriver 78.0.3904.105 (60e2d8774a8151efa6a00b1f358371b1e0e07ee2-refs/branch-heads/3904@{#877}) on port 20999

Only local connections are allowed.

Please protect ports used by ChromeDriver and related test frameworks to prevent access by malicious code.

[1576037205.886][WARNING]: Timed out connecting to Chrome, retrying...

Dec 10, 2019 11:06:48 PM org.openqa.selenium.remote.ProtocolHandshake createSession

INFO: Detected dialect: W3C

Test #6 Passed!

Starting ChromeDriver 78.0.3904.105 (60e2d8774a8151efa6a00b1f358371b1e0e07ee2-refs/branch-heads/3904@{#877}) on port 8242

Only local connections are allowed.

Please protect ports used by ChromeDriver and related test frameworks to prevent access by malicious code.

[1576037228.792][WARNING]: Timed out connecting to Chrome, retrying...

Dec 10, 2019 11:07:11 PM org.openqa.selenium.remote.ProtocolHandshake createSession

INFO: Detected dialect: W3C

Test #7 Passed!
Starting ChromeDriver 78.0.3904.105 (60e2d8774a8151efa6a00b1f358371b1e0e07ee2-refs/branch-heads/3904@{#877}) on port 12065
Only local connections are allowed.
Please protect ports used by ChromeDriver and related test frameworks to prevent access by malicious code.
[1576037251.366][WARNING]: Timed out connecting to Chrome, retrying...
Dec 10, 2019 11:07:33 PM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: W3C
Test #8 Passed!
All Tests Completed