



200

Oracle Public Cloud Workshop

Cloud Native Continuous
Delivery of Java Microservices

Contact: Dennis Foley dennis.foley@oracle.com

November 17, 2016

Introduction

This is the second of several labs that are part of the **Oracle Public Cloud DevOps Cloud Native Microservices** workshop. This workshop will walk you through the Software Development Lifecycle (SDLC) for a Cloud Native project that will create and use several Microservices.

In the first lab (100), the Project Manager created a new project in the Developer Cloud Service, added team members to the project, and created and assigned tasks to the developers of this application. In this lab, you will assume the persona of the Java developer, who will be tasked with creating several microservices that will supply data to any required front-end or analytics components (one of which you will build in the following lab, lab 300).

Please direct comments to: Dennis Foley (dennis.foley@oracle.com)

Objectives

- Access Developer Cloud Service
- Import Code from external Git Repository
- Import Project into Eclipse
- Build and Deploy project using Developer Cloud Service and Oracle Application Container Cloud Service

Required Artifacts

- The following lab requires an Oracle Public Cloud account that will be supplied by your instructor. You will need to download and install latest version of Eclipse

Outline

Introduction	2
Objectives	2
Required Artifacts.....	2
Outline.....	3
Create Initial Static Twitter Feed Service	4
Explore Developer Cloud Service	4
Create Initial Git Repository.....	11
Create Default Build and Deployment Process	14
Verify Twitter Feed Microservice deployment	23
Add Filter to Static Twitter Feed Service.....	28
Clone Project to Eclipse IDE.....	28
Test the Local Cloned Services.....	36
Add the Filter to the Service	40
Test the Local Filtered Services	43
Create a new Branch and Commit Code	44
Create a Branch and Commit Code	44
Create Merge Request.....	47
Merge the Branch as Lisa Jones	49
Test the JavaTwitterMicroservice in the Cloud	56
Supplementary Assignment – Twitter Live Feed Credentials	59
Create Twitter App	59
Appendix 2 – Installing Eclipse.....	68
Download and Install Eclipse	68
Optionally Configure Proxies	68

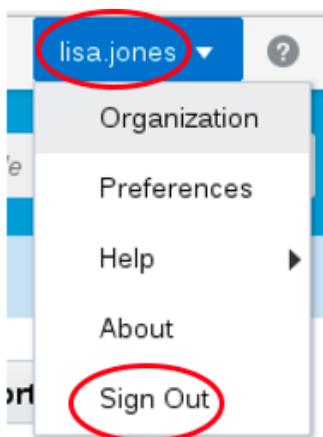
Create Initial Static Twitter Feed Service

Explore Developer Cloud Service

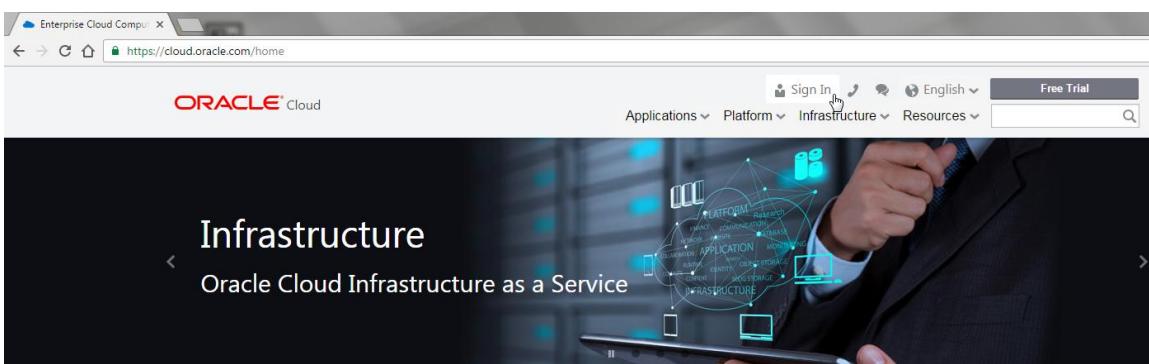
Note: If you are **NOT** running this lab with the multiple users (e.g. Lisa.Jones, Bala.Gupta and John.Dunbar), then skip past Steps 1, 2, 3 and 4, and go to Step 5. You will however logically assume the persona of the Developer Bala.Gupta, even though you remain logged in as the same user with which you completed lab 100.

STEP 1: Login to your Oracle Cloud account as Bala.Gupta

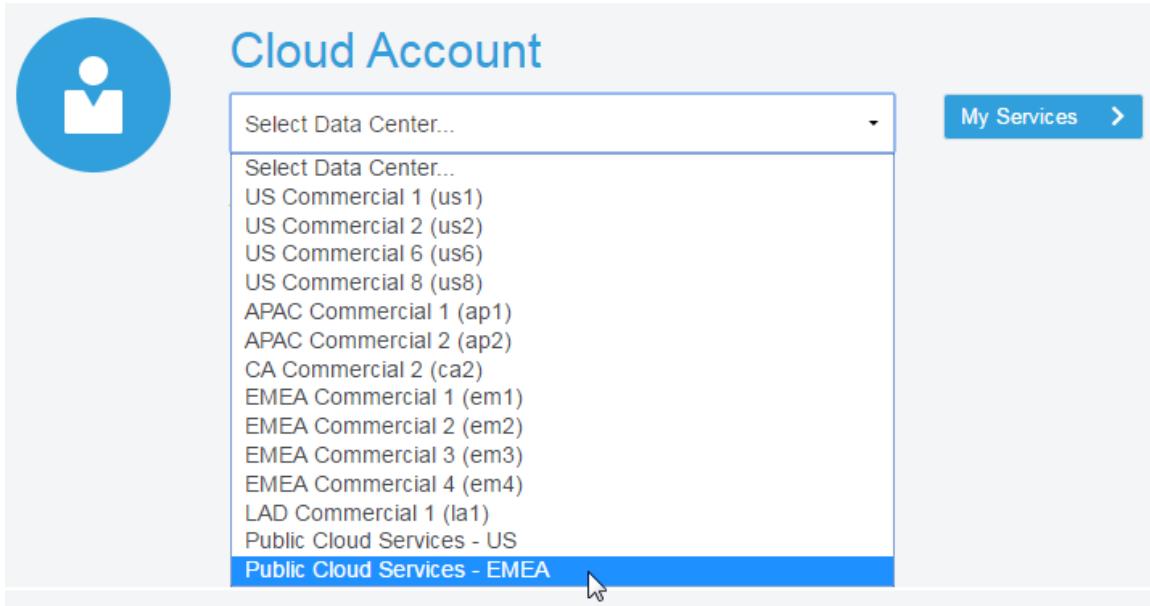
- If you just completed lab 100, or if you are still logged in as Lisa.Jones, you will need to first sign out before continuing this lab. Sign out by clicking on the user's name (lisa.jones) at the top right corner of the screen, then selecting **Sign Out** from the dropdown menu.



- Now we can login again. From any browser, go to the following URL:
<https://cloud.oracle.com>
- Click **Sign In** in the upper right hand corner of the browser.



- IMPORTANT** - Under My Services, ask your instructor which **Region** to select from the drop down list, and click on the **My Services** button.



- Enter your identity domain and click **Go**

NOTE: the **Identity Domain**, **User Name** and **Password** values will be given to you from your instructor.

The screenshot shows a form for entering an identity domain. It features a blue triangular graphic on the left. To its right is the text "Enter your Identity Domain" followed by a text input field containing the value "gse00002057". Below the input field is a blue button labeled "Go".

- Once your Identity Domain is set, enter your User Name and Password and click **Sign In**

NOTE: For the first part of this lab you will be acting as the Java Developer **Bala Gupta**. As with the previous lab, if you are not able to support multiple users, login as a supported user, and assume the "logical" identity of Bala Gupta - the Java Developer.

Welcome gse00002057 [change domain](#) [?](#)

[Can't access your account?](#)

- Once connected, you will be presented with a Dashboard displaying the various cloud services available to this account. Note: Based on your browser cookie settings, it's possible that once connected you will be placed into the Developer Cloud Service dashboard, and will not need to complete the next few tasks that were performed automatically.

Account gse00002055 [?](#)

0 Important Notifications

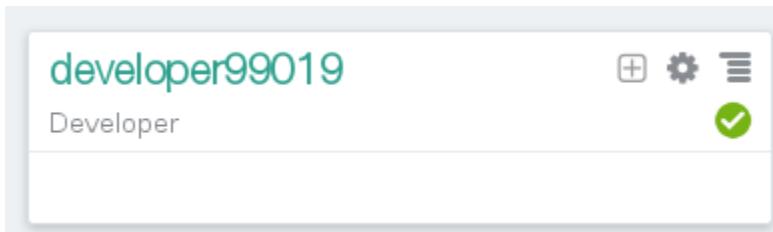
Dashboard [Create Instance](#) [Customize Dashboard](#)

developer99019 Developer	Database Instances	Compute Instances	Java Instances
Application Container Instances	Database Backup	Storage	SOA

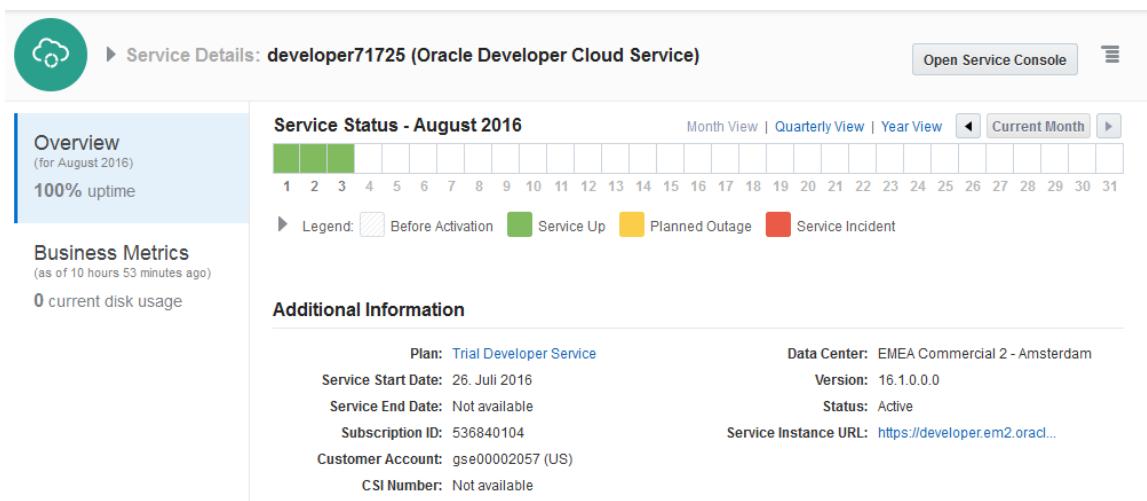
STEP 2: Login to Developer Cloud Service

Oracle Developer Cloud Service provides a complete development platform that streamlines team development processes and automates software delivery. The integrated platform includes issue tracking system, agile development dashboards, code versioning and code review platform, continuous integration and delivery automation, as well as team collaboration features such as wikis and live activity stream. With a rich web based dashboard and integration with popular development tools, Oracle Developer Cloud Service helps deliver better applications faster.

- From Cloud UI dashboard click on the **Developer** service. In this example the Developer Cloud Service is named **developer99019**.



- The Service Details page gives you a quick glance of the service status. Click **Open Service Console** for the Oracle Developer Cloud Service



- The Service Console will list all projects that you are currently a member of. Click **Twitter Feed Marketing Project** to access the project

The screenshot shows the Oracle Service Console interface. At the top, there are three tabs: 'Member' (highlighted in blue), 'Favorites', and 'All'. To the right of these is a green button with a plus sign and the text '+ New Project'. Below the tabs is a search bar with the placeholder 'Filter Projects' and a magnifying glass icon. The main content area displays a project card for 'Twitter Feed Marketing Project', which includes the project name, a brief description 'Project to gather and analyze twitter data', and a blue star icon.

- The Twitter Feed Marketing Project dashboard will be displayed.

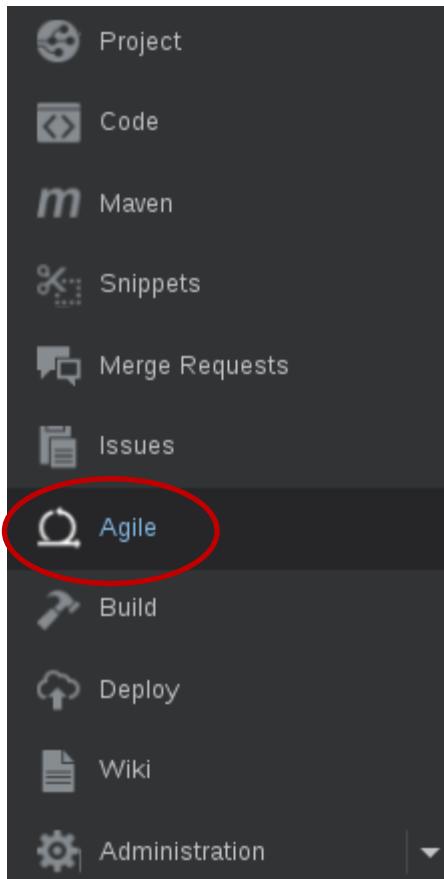
The screenshot shows the Oracle Developer Cloud Service dashboard for the 'Twitter Feed Marketing Project'. The left sidebar contains links for Project, Code, Maven, Snippets, Merge Requests, Issues, Agile, Build, Deploy, Wiki, and Administration. The main area displays a timeline of recent activities:

- Lisa Jones created Feature 4: Display Twitter Feed in Table Format
- Lisa Jones created Task 3: Create Initial GIT Repository for Twitter Feed Marketing UI
- Lisa Jones created Feature 2: Create Filter on Twitter Feed
- Lisa Jones created Task 1: Create Initial GIT Repository for Twitter Feed Service
- Lisa Jones assigned John Dunbar to the project as owner
- Lisa Jones assigned Bala Gupta to the project as owner
- Project created by Lisa Jones

A modal window titled 'Using DevCS with Oracle Database and APEX' is open, providing instructions on how to use DevCS version control and continuous integration for APEX applications. The 'REPOSITORIES' section shows a message: 'There are no git repositories to display. No git repositories exist in this project.'

STEP 3: Review Agile Board

- Within the **Twitter Feed Marketing Project**, click on **Agile** found on the left hand navigation.



STEP 4: Show Microservices Board

- If the **Microservices** list is not displayed as shown below, then click on the **Board Dropdown**, select **All**, and click on **Microservices**.

The screenshot shows two Jira boards side-by-side.

Top Board (Correct): This board is titled "Twitter Feed Marketing Project". It displays a sprint backlog for "Sprint 1 - Initial Development" with 4 issues. A red box highlights the "Microservices" dropdown menu at the top left of the board area. A red arrow points from the text "Correct" to this highlighted area.

Issue Type	Priority	Summary
Feature	!!	Feature 4: Display Twitter Feed in Table Format
Task	!!	Task 3: Create Initial GIT Repository for Twitter Feed Marketing UI
Feature	!!	Feature 2: Create Filter on Twitter Feed
Task	!!	Task 1: Create Initial GIT Repository for Twitter Feed Service

Bottom Board (Requires Change): This board is titled "System Board". It shows a list of boards under "All". A red box highlights the "System Board" dropdown menu at the top left. A red box highlights the "All" button in the navigation bar, which is also highlighted by a red arrow. Another red arrow points from the text "Requires Change" to the "Microservices" board entry in the list.

Board Name	Description	Star Icon
Microservices	No description	Star icon
System Board	System board	Star icon

STEP 5: Display the Active Sprints

- Click on the **Microservices** Board **Active Sprints**.

The screenshot shows the Microservices board interface. At the top, there's a navigation bar with tabs: Backlog, **Active Sprints** (which is highlighted with a red circle), Reports, and Board. Below the navigation bar, there's a search bar and some filters: Sort Issues by: Priority and Sort Swimlanes by: User Name. The main area is divided into four swim-lanes: To Do (4 items), In Progress (0 items), Verify Code (0 items), and Completed (0 items). Under the To Do swim-lane, there are two entries for user bala.gupta: Feature 2 (2 items) and Task 1 (1 item), both labeled 'UNCONFIRMED'. Under the In Progress swim-lane, there are two entries for user john.dunbar: Feature 4 (2 items) and Task 3 (1 item), also labeled 'UNCONFIRMED'.

Create Initial Git Repository

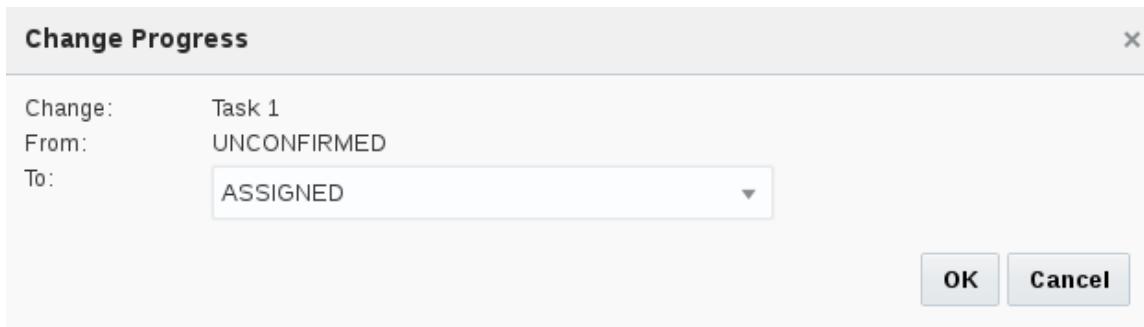
STEP 6: Create Initial Git Repository

To begin development on our Twitter feed microservices, we could start coding from scratch. However, prior to the formal kickoff of this project, you have already started doing some proof-of-concept development outside of the Developer Cloud Service in order to assess the feasibility of your assignment. You want to bring that existing code into the Developer Cloud Service as a starting point for your microservices. You will do that by cloning your external GIT repository into the Developer Cloud Service. Your first step will be to accept your task using the agile board.

- Drag and drop **Task1 - Create Initial GIT Repository for Twitter Feed Service** into the **In Progress** swim-lane.

The screenshot shows the Microservices board interface again. The 'Active Sprints' tab is selected. The board shows tasks for user bala.gupta. A red dashed box surrounds the 'To Do (4)' section, and a red arrow points from the 'Task 1' entry (which is part of this box) to the 'In Progress (0)' section. The other sections (Verify Code, Completed) are empty.

- Click **OK** on Change Progress popup.



The screenshot shows a dashboard interface. At the top, there are two summary boxes: "To Do (3)" with a count of 5 and "In Progress (1)" with a count of 1. Below these are two task cards under the heading "bala.gupta". The first card is for "Feature 2" with the sub-task "Create Filter on Twitter Feed", status "UNCONFIRMED", and a progress bar showing 2 completed tasks. The second card is for "Task 1" with the sub-task "Create Initial GIT Repository for Twitter Fe...", status "ASSIGNED", and a progress bar showing 1 completed task.

- In the left hand navigation panel, click **Project**
- Click on **New Repository** to create a new Git Repository

The screenshot shows the "REPOSITORIES" section of a project page. At the top, there is a green button labeled "+ New Repository". Below it is a search bar with the placeholder "Filter Git Repositories" and a magnifying glass icon. Underneath the search bar are two buttons: "All" (which is blue and selected) and "Favorites". To the right of these buttons is a dropdown menu labeled "Favorites First". In the center, there is a large circular icon with a diagonal slash over a document, indicating that no repositories are present. Below this icon, the text "There are no git repositories to display" and "No git repositories exist in this project." is displayed. At the bottom of the section, there is a "Maven" section with tabs for "HTTP", "DAV", and a URL "https://developer.em2.oraclecloud.com/profile/developer71725-gse00002057/s/developer7172". On the right side of the page, there is a vertical sidebar with three icons: a list, a chart, and a user profile.

- In the New Repository wizard enter the following information and click **Create**.

```
Name = TwitterFeedMicroservice  
Description = Twitter Feed Microservice  
Initial content = Import existing repository  
and enter the URL:  
https://github.com/oraclenassolutionengineering/TwitterFeed.git
```

New Repository

* Name	TwitterFeedMicroservice
Description	Twitter Feed Microservice
Initial content	<input checked="" type="radio"/> Empty Repository <input type="radio"/> Initialize repository with README file <input checked="" type="radio"/> Import existing repository https://github.com/oraclenassolutionengineering/TwitterFeed.git
► Credentials for non-public repos	
Create Cancel	

- You have now created a new GIT repository based on an existing repository.

The screenshot shows a Git commit history for a repository named "TwitterFeedMicroservice.git" on the "master" branch. The commits are listed below:

File	Commit Message
.settings	fixed Eclipse project issues, updated maven plugin versions, added JUnit philchung
src	Added , and CR to the end of the record in the Tweets list oraclenassolutionengineering
.classpath	Modified code to support the Static Tweets in the SampleStream, pom, oracle
.DS_Store	moved everything up a directory and removed jersey-example philchung
.gitignore	moved everything up a directory and removed jersey-example philchung
.project	moved everything up a directory and removed jersey-example philchung
build.sh	moved everything up a directory and removed jersey-example philchung
DEPLOY.md	moved everything up a directory and removed jersey-example philchung
deploy.sh	moved everything up a directory and removed jersey-example philchung
manifest.json	moved everything up a directory and removed jersey-example philchung
pom.xml	Modified code to support the Static Tweets in the SampleStream, pom, oracle
README.md	Updated README.md philchung

Create Default Build and Deployment Process

STEP 7: Create Default Build Process

Now that we have the source code in our managed GIT repository, we need to create a build process that will be triggered whenever a commit is made to the master branch. We will set up a Maven build process in this section.

- On navigation panel click **Build** to access the build page and click **New Job**.



 New Job

- In the New Job popup enter **Twitter Feed Build** for the Job Name, and then click **Save**.

New Job

* Job Name Twitter Feed Build

Create a free-style job
 Copy existing job

Save **Cancel**

- You are now placed into the job configuration screen.

◀ Jobs Overview | Twitter Feed Build | Configure build job

Main Build Parameters Source Control Triggers Environment Build Steps Post Build Advanced

* Name Twitter Feed Build

Description

JDK Default (The default Java version in the executing environment)

Disable build
 Execute concurrent builds if necessary
 Discard old builds

- On the Main tab of the Configure Build screen change the **JDK** drop down to **JDK8**.



- Click the **Source Control** tab. Click **Git** and select the **TwitterFeedMicroservice.git** from the drop down.

[◀ Jobs Overview](#) | [Twitter Feed Build](#) | [Configure build job](#)

Main Build Parameters **Source Control** Triggers Environment Build Steps Post Build Advanced

To integrate the Build System with Source Control, select an option below and then configure the required settings.

- None
 Git

Repositories

Add

* Repository

TwitterFeedMicroservice.git

x

► Advanced Repository Settings

Branches

Add

► Advanced Git Settings

- Click the **Triggers** tab. Select **Based on SCM polling schedule**, and add a schedule: ***/1******

Note: The above expression results in the repository being polled every minute to check for any changes. If there are changes, the build will trigger.

[◀ Jobs Overview](#) | [Twitter Feed Build](#) | [Configure build job](#)

Main Build Parameters Source Control Triggers **Triggers** Environment Build Steps Post Build Advanced

When these jobs are built
 Based on this schedule
 Based on SCM polling schedule

Schedule `*/1***`

Polling will automatically trigger on commits to the SCM. Specify a schedule only for any additional polling required.

When Maven dependencies have been updated by Maven 3 integration

- Click the **Build Steps** tab. Click **Add Build Step**, and select **Invoke Maven 3**.

[◀ Jobs Overview](#) | [Twitter Feed Build](#) | [Configure build job](#)

Main Build Parameters Source Control Triggers Environment **Build Steps** Post Build Advanced

Configure Build Steps

Add Build Step ▾

- Execute shell
- Invoke Ant
- Invoke Maven 2 (Legacy)
- Invoke Maven 3** (highlighted with a red oval)
- Invoke Gradle
- Invoke NodeJS
- Copy Artifacts

- Change **Goals** to `clean assembly:assembly`

[◀ Jobs Overview](#) | [Twitter Feed Build](#) | [Configure build job](#)

Main Build Parameters Source Control Triggers Environment **Build Steps** Post Build Advanced

Configure Build Steps

Add Build Step ▾

Invoke Maven 3

Maven 3 (Bundled) ▼ X

Goals `clean assembly:assembly`

- Click the **Post Build** tab. Check **Archive the artifacts** and enter ****/target/*** for Files to Archive. Verify **GZIP** in the Compression Type. Also, check **Publish JUnit test report** and enter ****/target/surefire-reports/*.xml** for the Test Report XMLs. This will provide a report on the Test Scripts results for each build.

< Jobs Overview | Twitter Feed Build | Configure build job

Main Build Parameters Source Control Triggers Environment Build Steps **Post Build** Advanced

Aggregate downstream test results
 Build other jobs
 Archive the artifacts

* **Files To Archive** `**/target/*`
 Enable auto validation for file masks

Excludes
 Discard all but the last successful/stable artifact to save disk space

Compression Type GZIP

Record fingerprints of files to track usage
 Publish Javadoc
 Publish JUnit test result report

Test Report XMLs `**/target/surefire-reports/*.xml`
 Retain long standard output/error

Archive Maven 3 artifacts
 Record fingerprints of Maven artifacts
 Git publisher
 Notify that Maven dependencies have been updated by Maven 3 integration
 Oracle Cloud Service Deployment

- Click **Save** to complete the configuration.
- Click the **Build Now** button to start the build immediately. Wait, as it may take 30 seconds or more, but the status will change to the following:

[◀ Jobs Overview](#) | Twitter Feed Build

Description

No description available

Permalinks

[Last](#) | Successful | Failed | Completed | Unsuccessful | Stable | Unstable

Notifications

On Off

CC Me

Build History

New Build

 N/A

Status	Build	Time	Duration	Console
	#1	Just now	N/A	

- Once the build begins, it should take about approximately 1 to 2 minutes for the build to complete. Once complete, you will be able to see the number of successful test runs in the Test Result Trend section. Wait for the build to complete before continuing to the next step, as we need the build artifact to complete the deployment configuration. After the build begins, you can also click on the **Console Icon** to monitor the build log details.

[◀ Jobs Overview](#) | Twitter Feed Build

Build Now | Configure | Disable | Delete

Description
No description available

Permalinks
[Last](#) | [Successful](#) | Failed | [Completed](#) | [Unsuccessful](#) | [Stable](#) | [Unstable](#)

Notifications On Off CC Me

Build History

Status	Build	Time	Duration	Console
	#1	Just now	1 min 51 s	

Artifacts of Last Successful Build

-  target
-  (all files in zip)

Build Trend



Build duration in seconds

120
100
80
60
40
20
0

Build Number

— Success

STEP 8: Create Default Deployment Process

Now that we have successfully built our project, we need to create a deployment configuration that will watch for stable builds and deploy them to a new Application Container Cloud Service instance for testing.

- On navigation panel click **Deploy** to access the Deployment page. Click **New Configuration**.



You currently have no deployment configurations.

Define a new configuration plan for deploying an application in your project to an Oracle cloud service instance.

 **New Configuration**

- Enter the following data:

```
Configuration Name = TwitterFeedMicroserviceDeploy
Application Name = JavaTwitterMicroservice
```

* Configuration Name

* Application Name

- To the Right of Deployment Target, click **New** and select **Application Container Cloud**

* Configuration Name	<input type="text" value="Twitter Feed Microservice Deploy"/>
* Application Name	<input type="text" value="JavaTwitterMicroservice"/>
* Deployment Target	<input type="text" value="Select a Java Service"/> <div style="margin-left: 10px; border: 1px solid #ccc; padding: 2px; display: inline-block;"> New ▾ </div> <div style="margin-left: 10px; border: 1px solid #ccc; padding: 2px; display: inline-block; background-color: #f0f0f0;"> Java Cloud Service - SaaS Extension... </div> <div style="margin-left: 10px; border: 1px solid #ccc; padding: 2px; display: inline-block; background-color: #f0f0f0;"> Java Cloud Service... </div> <div style="margin-left: 10px; border: 1px solid #cc0000; padding: 2px; display: inline-block; background-color: #ffcccc;"> Application Container Cloud... </div>
Type	<input checked="" type="radio"/> On Demand <input type="radio"/> Automatic
* Job	<input type="text" value="Twitter Feed Build"/>
* Build	<input type="text" value="1 (Success)"/>
* Artifact	<input type="text" value="Select an Artifact"/>

- Enter the following data and click **Test Connection**. If Successful, click **Use Connection**

```
Data Center = EMEA Commercial 2 - em2
  ** or your appropriate Data Center **
Identity Domain = <You Identity Domain>
Username = bala.gupta (or your appropriate username
                     if running as single user)
Password = <Supplied Password>
```

Deploy to Application Container Cloud

* Data Center	EMEA Commercial 2 - em2
* Identity Domain	gse00002057
* Username	bala.gupta
* Password	*****
✓ Successful Use Connection	

- Set ACCS Properties to Runtime **Java** and Subscription **Hourly**. Click **Automatic**. Select **target/twitter-microservice-example-dist.zip** for the **Artifact**.

ACCS Properties	Runtime <input checked="" type="radio"/> Java <input type="radio"/> Node	Subscription <input checked="" type="radio"/> Hourly <input type="radio"/> Monthly
Type	<input type="radio"/> On Demand <input checked="" type="radio"/> Automatic <input checked="" type="checkbox"/> Deploy stable builds only	
* Job	Twitter Feed Build	
* Artifact	target/twitter-microservice-example-dist.zip	

- Click **Save**

New Deployment Configuration

* Configuration Name	TwitterFeedMicroserviceDeploy	Save Cancel
* Application Name	JavaTwitterMicroservice	
* Deployment Target	em2 / gse00002057 / bala.gupta	New
ACCS Properties	Runtime <input checked="" type="radio"/> Java <input type="radio"/> Node	Subscription <input checked="" type="radio"/> Hourly <input type="radio"/> Monthly
Type	<input type="radio"/> On Demand <input checked="" type="radio"/> Automatic <input checked="" type="checkbox"/> Deploy stable builds only	
* Job	Twitter Feed Build	
* Artifact	target/twitter-microservice-example-dist.zip	

- Click drop down and select **Start**

The screenshot shows the Oracle Cloud Control interface with the 'Deployments' section selected. A deployment named 'JavaTwitterMicroservice' is listed. A context menu is open over this deployment, with the 'Start' option highlighted. Other options in the menu include 'Redeploy', 'Edit Configuration', and 'Delete Configuration'. The status of the deployment is shown as 'Succeeded'.

- Wait until the message **Starting application** changes to **Last deployment succeeded**

Deployments

The screenshot shows the Oracle Cloud Control interface with the 'Deployments' section selected. A deployment named 'JavaTwitterMicroservice' is listed. A message 'Starting application - Just now.' is displayed below the deployment details, with a red circle around it. Deployment details include: Deploy to ACCS em2 / gse00002055 / bala.gupta, Configuration TwitterFeedMicroserviceDeploy, Job / Build Twitter Feed Build / Latest Successful Build, Artifact target/twitter-microservice-example-dist.zip.

Deployments

The screenshot shows the Oracle Cloud Control interface with the 'Deployments' section selected. A deployment named 'JavaTwitterMicroservice' is listed. A message 'Last deployment succeeded - Just now.' is displayed below the deployment details, with a red circle around it. Deployment details are identical to the previous screenshot.

Verify Twitter Feed Microservice deployment

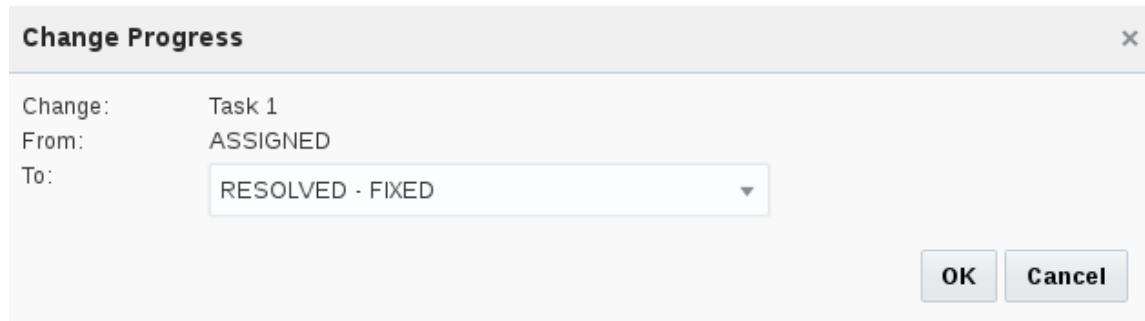
STEP 9: Change status to Verified

Now that we have successfully deployed the build artifact to the Application Container Cloud Service, we will update our agile board to reflect that status. Although the complexity of the next task (verification) is quite simple, we will still move the task to the "Verify Code" column before manually verifying the new functionality.

- On navigation panel click **Agile**, followed by clicking **Active Sprints**. Drag and drop **Task 1** from **In Progress** to the **Verify Code** column.



- In the Change Progress popup, click on **OK**

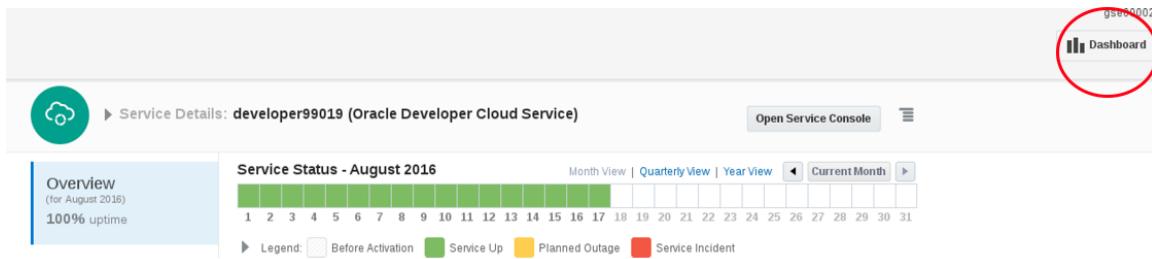


- The code is now ready for verification before moving to Completed

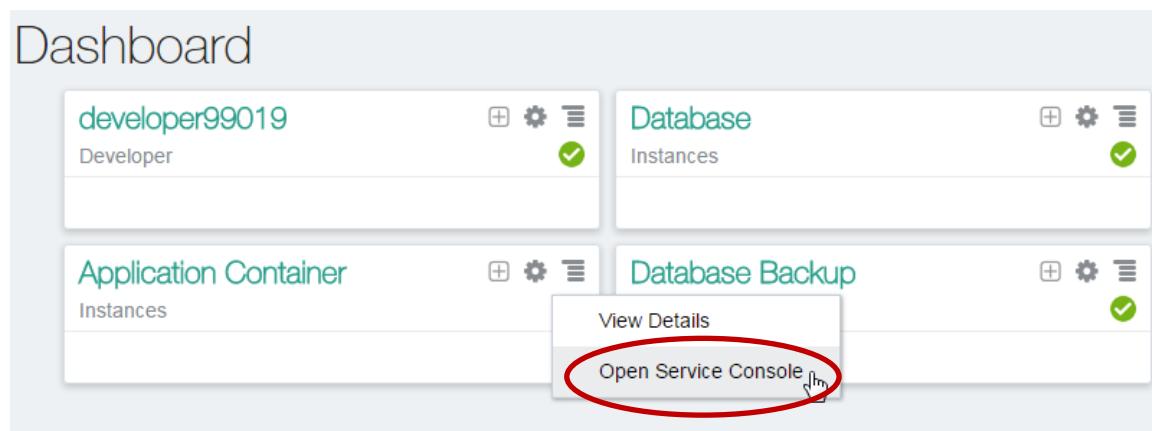


STEP 10: Login to Oracle Application Container Cloud Service

- Return to the Developer Service Cloud Dashboard tab if it's still available, then select the Dashboard icon to return to the Oracle Public Cloud Dashboard. Note: It's possible that you may be required to once again login, if the session has expired.



- Once the Oracle Public Cloud **Dashboard** is displayed, click on the  menu to the right of the **Application Container** service. Then select **Open Service Console**



- On the Application Container Cloud Service (ACCS) Service Console you can view all the deployed applications, including our newly created **JavaTwitterMicroservice**. Click on the **URL**, and it will load a new browser tab

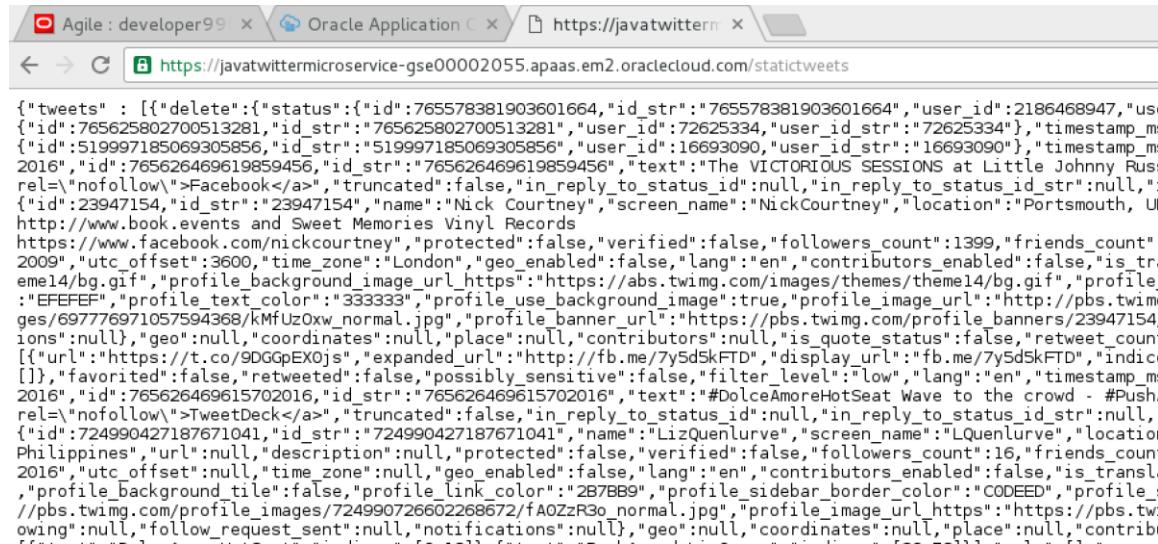
The screenshot shows the Oracle Public Cloud Applications page with the JavaTwitterMicroservice application listed. The URL link is highlighted with a red circle.

JavaTwitterMicroservice	Version: 1.0	Last Deployed On: Aug 15, 2016 8:34:05 PM UTC	Memory: 1GB
Java SE: 1.8.0_91-b14	Created On: Aug 15, 2016 8:34:05 PM UTC	Instances: 1	
URL: https://JavaTwitterMicroservice-gse00002057.apaas.em2.oraclecloud.com			

- Append **/statictweets** to the end of the URL in the browser, and press return.

```
https://javatwittermicroservice-
gse00002055.apaas.em2.oraclecloud.com/statictweets
```

- The URL should return a JSON array containing a Static Twitter feed. **Note:** If you desire to see a formatted view of the JSON, open a new tab and search Google for "JSONViewer chrome plugin" – After you install the Chrome Plugin and re-submit the URL, you will be able to view the JSON in a more readable format.



```
{"tweets" : [{"delete": {"status": {"id": 765578381903601664, "id_str": "765578381903601664", "user_id": 2186468947, "user_id_str": "2186468947", "user_id_hex": "3A8E0D", "user_id_hex_str": "3A8E0D", "user_name": "NickCourtney", "screen_name": "NickCourtney", "location": "Portsmouth, UK", "lang": "en", "contributors_enabled": false, "is_translator": false, "profile_image_url": "http://pbs.twimg.com/profile_images/23947154/nickcourtney_normal.jpg", "profile_banner_url": "https://pbs.twimg.com/profile_banners/23947154/1463125942", "profile_link_color": "#0080C0", "profile_sidebar_border_color": "#0080C0", "profile_sidebar_fill_color": "#F0F0F0", "profile_text_color": "#0080C0", "profile_use_background_image": true, "protected": false, "verified": false, "followers_count": 1399, "friends_count": 2009, "utc_offset": 3600, "time_zone": "London", "geo_enabled": false, "lang": "en", "contributors_enabled": false, "is_translator": false, "profile_image_url_https": "https://abs.twimg.com/images/themes/theme1/bg.gif", "profile_banner_url_https": "https://pbs.twimg.com/profile_banners/23947154/1463125942", "profile_link_color_https": "#0080C0", "profile_sidebar_border_color_https": "#0080C0", "profile_sidebar_fill_color_https": "#F0F0F0", "profile_text_color_https": "#0080C0", "profile_use_background_image_https": true, "description": "Sweet Memories Vinyl Records", "entities": {"hashtags": [], "symbols": [], "urls": [{"url": "http://www.facebook.com/nickcourtney", "expanded_url": "http://www.facebook.com/nickcourtney", "display_url": "Facebook", "indices": [0, 14]}], "user_mentions": []}, "created_at": "Mon Dec 01 14:45:42 +0000 2014", "text": "The VICTORIOUS SESSIONS at Little Johnny Russel's in Portsmouth, UK", "truncated": false, "in_reply_to_status_id": null, "in_reply_to_status_id_str": null, "in_reply_to_user_id": null, "in_reply_to_user_id_str": null, "in_reply_to_screen_name": null, "in_reply_to_user_name": null, "retweet_count": 0, "favorite_count": 0, "is_quote_status": false, "retweeted": false, "possibly_sensitive": false, "filter_level": "low", "lang": "en", "timestamp_ms": 1412408342000}, {"id": 765626469619859456, "id_str": "765626469619859456", "user_id": 16693090, "user_id_str": "16693090", "user_id_hex": "A6B30A", "user_id_hex_str": "A6B30A", "user_name": "ljohnnyrussell", "screen_name": "ljohnnyrussell", "location": "Little Johnny Russel's, Portsmouth, UK", "lang": "en", "contributors_enabled": false, "is_translator": false, "profile_image_url": "http://pbs.twimg.com/profile_images/23947154/ljohnnyrussell_normal.jpg", "profile_banner_url": "https://pbs.twimg.com/profile_banners/23947154/1463125942", "profile_link_color": "#0080C0", "profile_sidebar_border_color": "#0080C0", "profile_sidebar_fill_color": "#F0F0F0", "profile_text_color": "#0080C0", "profile_use_background_image": true, "protected": false, "verified": false, "followers_count": 1, "friends_count": 1, "utc_offset": null, "time_zone": null, "geo_enabled": false, "lang": "en", "contributors_enabled": false, "is_translator": false, "profile_image_url_https": "https://abs.twimg.com/images/themes/theme1/bg.gif", "profile_banner_url_https": "https://pbs.twimg.com/profile_banners/23947154/1463125942", "profile_link_color_https": "#0080C0", "profile_sidebar_border_color_https": "#0080C0", "profile_sidebar_fill_color_https": "#F0F0F0", "profile_text_color_https": "#0080C0", "profile_use_background_image_https": true, "description": "The VICTORIOUS SESSIONS at Little Johnny Russel's in Portsmouth, UK", "entities": {"hashtags": [], "symbols": [], "urls": [{"url": "http://www.facebook.com/ljohnnyrussell", "expanded_url": "http://www.facebook.com/ljohnnyrussell", "display_url": "Facebook", "indices": [0, 14]}], "user_mentions": []}, "created_at": "Mon Dec 01 14:45:42 +0000 2014", "text": "The VICTORIOUS SESSIONS at Little Johnny Russel's in Portsmouth, UK", "truncated": false, "in_reply_to_status_id": null, "in_reply_to_status_id_str": null, "in_reply_to_user_id": null, "in_reply_to_user_id_str": null, "in_reply_to_screen_name": null, "in_reply_to_user_name": null, "retweet_count": 0, "favorite_count": 0, "is_quote_status": false, "retweeted": false, "possibly_sensitive": false, "filter_level": "low", "lang": "en", "timestamp_ms": 1412408342000}, {"id": 765626469615702016, "id_str": "765626469615702016", "user_id": 765626469615702016, "user_id_str": "765626469615702016", "user_id_hex": "A6B30A", "user_id_hex_str": "A6B30A", "user_name": "DolceAmoreHotSeat", "screen_name": "DolceAmoreHotSeat", "location": "Portsmouth, UK", "lang": "en", "contributors_enabled": false, "is_translator": false, "profile_image_url": "http://pbs.twimg.com/profile_images/23947154/DolceAmoreHotSeat_normal.jpg", "profile_banner_url": "https://pbs.twimg.com/profile_banners/23947154/1463125942", "profile_link_color": "#0080C0", "profile_sidebar_border_color": "#0080C0", "profile_sidebar_fill_color": "#F0F0F0", "profile_text_color": "#0080C0", "profile_use_background_image": true, "protected": false, "verified": false, "followers_count": 1, "friends_count": 1, "utc_offset": null, "time_zone": null, "geo_enabled": false, "lang": "en", "contributors_enabled": false, "is_translator": false, "profile_image_url_https": "https://abs.twimg.com/images/themes/theme1/bg.gif", "profile_banner_url_https": "https://pbs.twimg.com/profile_banners/23947154/1463125942", "profile_link_color_https": "#0080C0", "profile_sidebar_border_color_https": "#0080C0", "profile_sidebar_fill_color_https": "#F0F0F0", "profile_text_color_https": "#0080C0", "profile_use_background_image_https": true, "description": "Wave to the crowd - #PushRelax", "entities": {"hashtags": [{"text": "#PushRelax"}], "symbols": [], "urls": [{"url": "http://www.pushrelax.com", "expanded_url": "http://www.pushrelax.com", "display_url": "PushRelax", "indices": [0, 14]}], "user_mentions": []}, "created_at": "Mon Dec 01 14:45:42 +0000 2014", "text": "Wave to the crowd - #PushRelax", "truncated": false, "in_reply_to_status_id": null, "in_reply_to_status_id_str": null, "in_reply_to_user_id": null, "in_reply_to_user_id_str": null, "in_reply_to_screen_name": null, "in_reply_to_user_name": null, "retweet_count": 0, "favorite_count": 0, "is_quote_status": false, "retweeted": false, "possibly_sensitive": false, "filter_level": "low", "lang": "en", "timestamp_ms": 1412408342000}, {"id": 724990427187671041, "id_str": "724990427187671041", "user_id": 724990427187671041, "user_id_str": "724990427187671041", "user_id_hex": "A6B30A", "user_id_hex_str": "A6B30A", "user_name": "LQuenlurve", "screen_name": "LQuenlurve", "location": "Philippines", "lang": "en", "contributors_enabled": false, "is_translator": false, "profile_image_url": "http://pbs.twimg.com/profile_images/23947154/LQuenlurve_normal.jpg", "profile_banner_url": "https://pbs.twimg.com/profile_banners/23947154/1463125942", "profile_link_color": "#0080C0", "profile_sidebar_border_color": "#0080C0", "profile_sidebar_fill_color": "#F0F0F0", "profile_text_color": "#0080C0", "profile_use_background_image": true, "protected": false, "verified": false, "followers_count": 16, "friends_count": 2009, "utc_offset": null, "time_zone": null, "geo_enabled": false, "lang": "en", "contributors_enabled": false, "is_translator": false, "profile_image_url_https": "https://abs.twimg.com/images/themes/theme1/bg.gif", "profile_banner_url_https": "https://pbs.twimg.com/profile_banners/23947154/1463125942", "profile_link_color_https": "#0080C0", "profile_sidebar_border_color_https": "#0080C0", "profile_sidebar_fill_color_https": "#F0F0F0", "profile_text_color_https": "#0080C0", "profile_use_background_image_https": true, "description": "Sweet Memories Vinyl Records", "entities": {"hashtags": [{"text": "#PushRelax"}], "symbols": [], "urls": [{"url": "http://www.pushrelax.com", "expanded_url": "http://www.pushrelax.com", "display_url": "PushRelax", "indices": [0, 14]}], "user_mentions": []}, "created_at": "Mon Dec 01 14:45:42 +0000 2014", "text": "Sweet Memories Vinyl Records", "truncated": false, "in_reply_to_status_id": null, "in_reply_to_status_id_str": null, "in_reply_to_user_id": null, "in_reply_to_user_id_str": null, "in_reply_to_screen_name": null, "in_reply_to_user_name": null, "retweet_count": 0, "favorite_count": 0, "is_quote_status": false, "retweeted": false, "possibly_sensitive": false, "filter_level": "low", "lang": "en", "timestamp_ms": 1412408342000}, {"id": 724990726602268672, "id_str": "724990726602268672", "user_id": 724990726602268672, "user_id_str": "724990726602268672", "user_id_hex": "A6B30A", "user_id_hex_str": "A6B30A", "user_name": "fAOZzR3o", "screen_name": "fAOZzR3o", "location": "Portsmouth, UK", "lang": "en", "contributors_enabled": false, "is_translator": false, "profile_image_url": "http://pbs.twimg.com/profile_images/23947154/fAOZzR3o_normal.jpg", "profile_banner_url": "https://pbs.twimg.com/profile_banners/23947154/1463125942", "profile_link_color": "#0080C0", "profile_sidebar_border_color": "#0080C0", "profile_sidebar_fill_color": "#F0F0F0", "profile_text_color": "#0080C0", "profile_use_background_image": true, "protected": false, "verified": false, "followers_count": 1, "friends_count": 1, "utc_offset": null, "time_zone": null, "geo_enabled": false, "lang": "en", "contributors_enabled": false, "is_translator": false, "profile_image_url_https": "https://abs.twimg.com/images/themes/theme1/bg.gif", "profile_banner_url_https": "https://pbs.twimg.com/profile_banners/23947154/1463125942", "profile_link_color_https": "#0080C0", "profile_sidebar_border_color_https": "#0080C0", "profile_sidebar_fill_color_https": "#F0F0F0", "profile_text_color_https": "#0080C0", "profile_use_background_image_https": true, "description": "Sweet Memories Vinyl Records", "entities": {"hashtags": [{"text": "#PushRelax"}], "symbols": [], "urls": [{"url": "http://www.pushrelax.com", "expanded_url": "http://www.pushrelax.com", "display_url": "PushRelax", "indices": [0, 14]}], "user_mentions": []}, "created_at": "Mon Dec 01 14:45:42 +0000 2014", "text": "Sweet Memories Vinyl Records", "truncated": false, "in_reply_to_status_id": null, "in_reply_to_status_id_str": null, "in_reply_to_user_id": null, "in_reply_to_user_id_str": null, "in_reply_to_screen_name": null, "in_reply_to_user_name": null, "retweet_count": 0, "favorite_count": 0, "is_quote_status": false, "retweeted": false, "possibly_sensitive": false, "filter_level": "low", "lang": "en", "timestamp_ms": 1412408342000}], "meta": {"total": 5}
```

STEP 11: Complete Task

We have now verified that the statictweets microservice has been deployed and functions properly. To finish up this part of the lab, we will mark the Issue as completed in our Sprint.

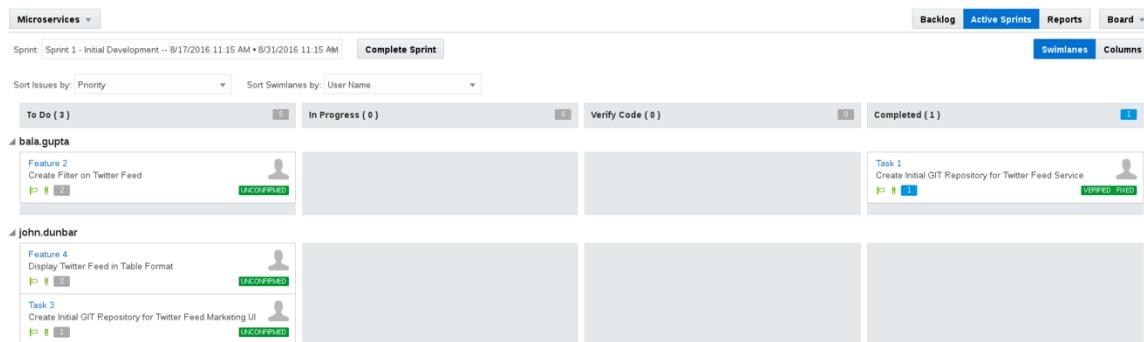
- Back in the Developer Cloud Service window, click **Agile**, followed by clicking **Active Sprints**.
- Drag and drop **Task 1** from **Verify Code** to **Completed**.



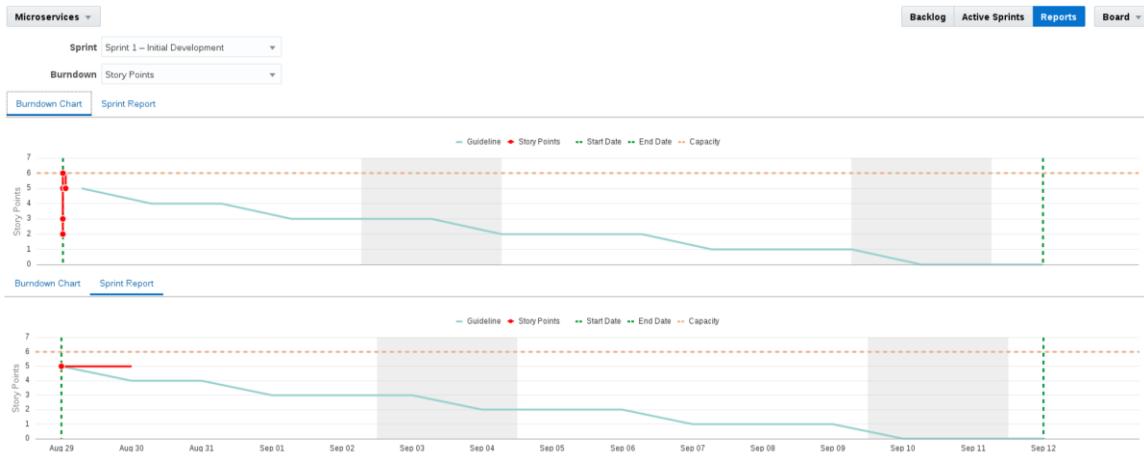
- In the Change Progress popup click **OK**.



- Your Sprint should now look like the following:



- You can also click on the **Reports** button and view your progress in the **Burndown Chart** and **Sprint Report**.



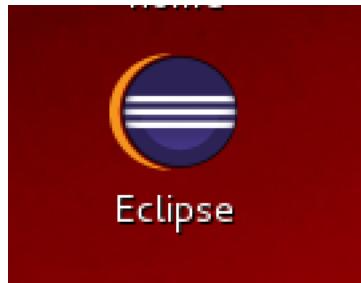
Add Filter to Static Twitter Feed Service

Now that we have completed the import, build, deployment, and verification of our initial static twitter microservice, it is time to extend the project by adding a new microservice that allows us to dynamically filter the incoming tweets based on their contents. We will use the Eclipse IDE to clone the managed GIT repository to our local workstation, test the local copy, and add the filtering feature to the local copy. We will test the new feature in Eclipse, create a new code branch for it, and commit the branch. Then we will create a merge request and switch to the Project Manager persona to approve that request. We will also see how we can manage our agile task status directly from Eclipse.

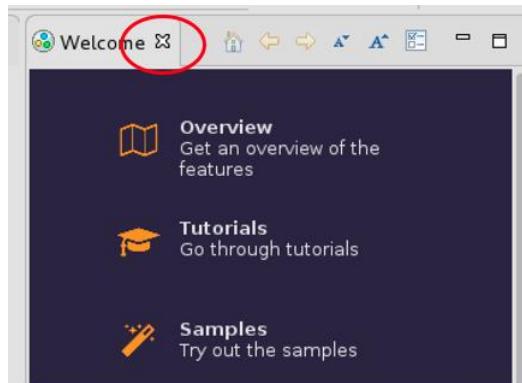
Clone Project to Eclipse IDE

STEP 12: Load Eclipse IDE

- Right Click and select **Run** on the **Eclipse** Desktop Icon

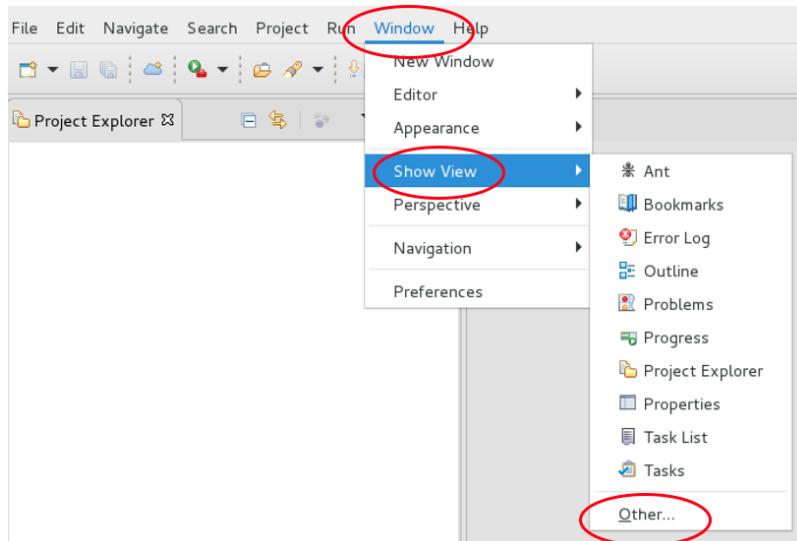


- Once Eclipse loads, **close** the **Welcome** Window if it is visible.

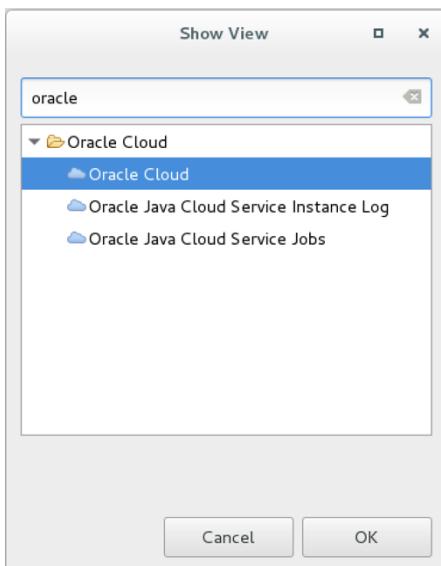


STEP 13: Create connection to Oracle Developer Cloud Service

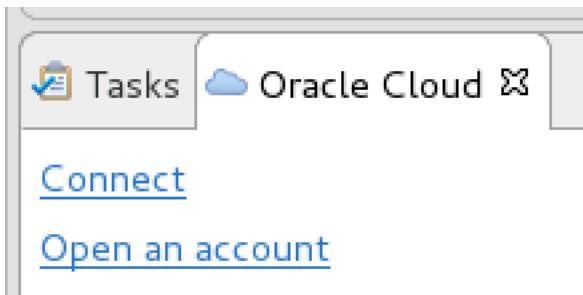
- We will now create a connection to the Developer Cloud Service. To do this, first click on the menu options **Window -> Show View ->Other**



- Enter **oracle** in the search field. Select **Oracle Cloud**, and click on **OK**.

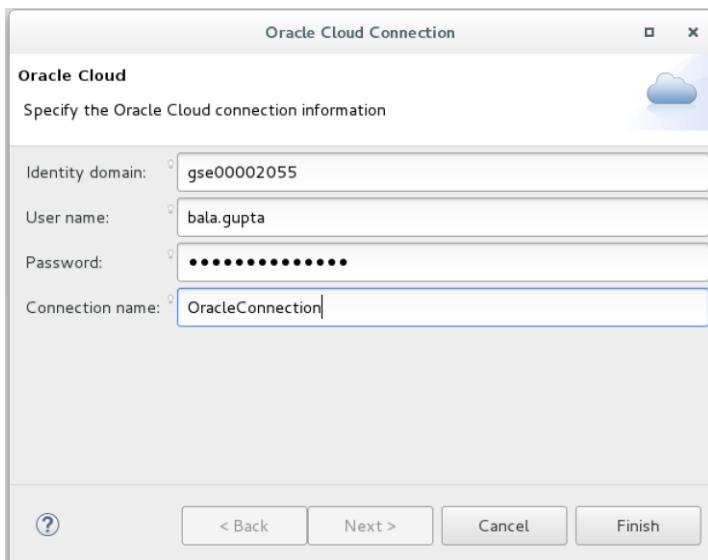


- Click on **Connect** in the Oracle Cloud tab



- Enter the following information:

```
Identity Domain: <your identity domain>
User name:      bala.gupta (or your appropriate username,
                  if running as single user)
Password:       <your Identity domain password>
Connection Name: OracleConnection
```



- If prompted, enter and confirm a Master Password for the Eclipse Secure Storage. In our example we use the **password** of **oracle**. Next, press **OK**.

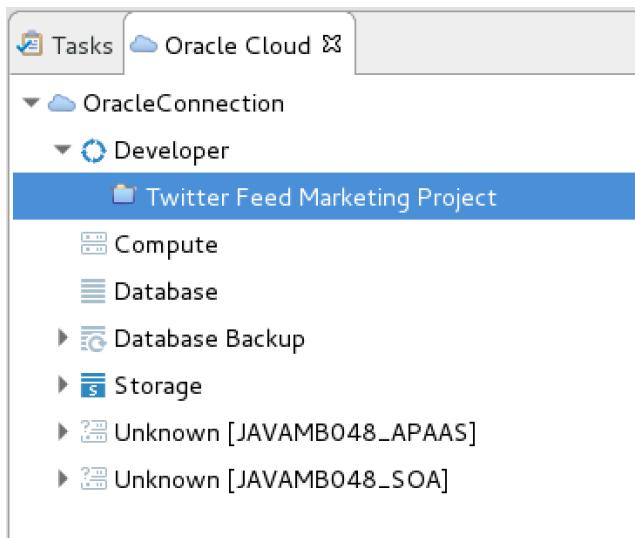


- If prompted to enter a Password Hint, click on **No**

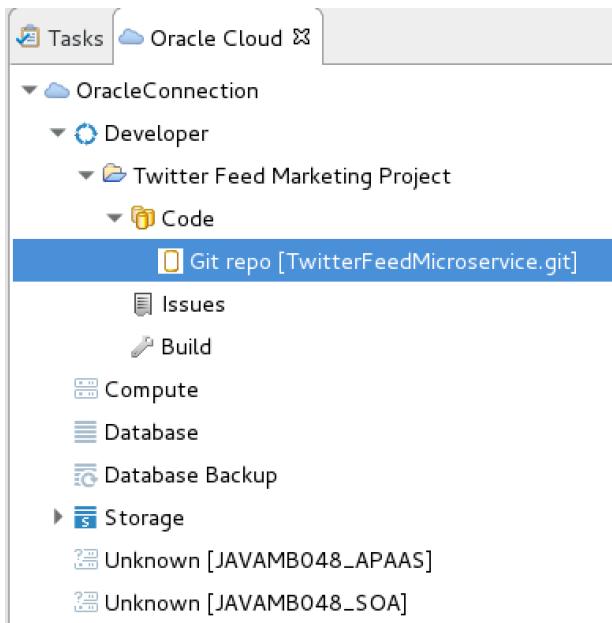


STEP 14: Create a local clone of the repository

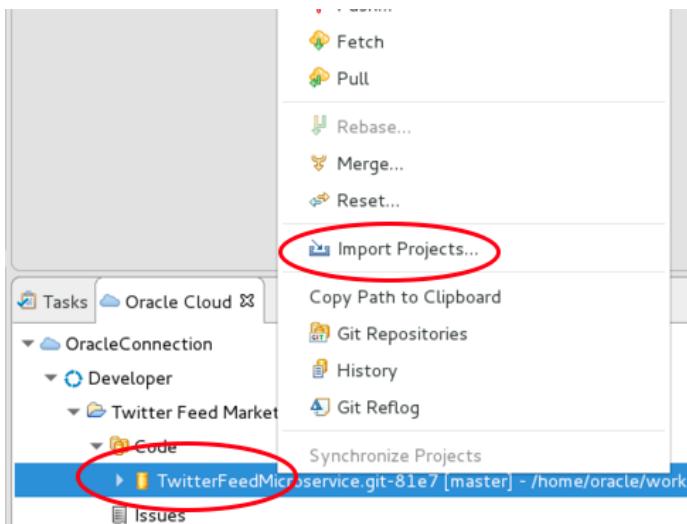
- Expand Developer**, and then **double click** on **Twitter Feed Marketing Project** to activate the project.



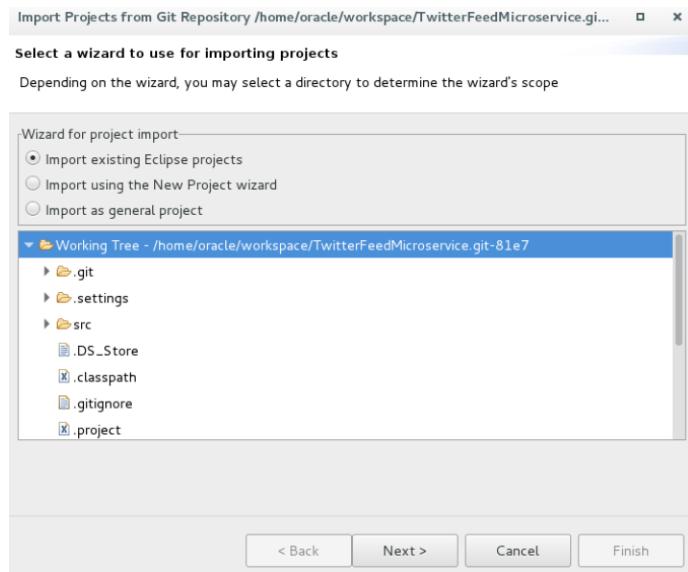
- Expand the Code**, and double click on the **Git Repo [TwitterFeedMicroservice.git]**, to cause the Repo to be cloned locally.



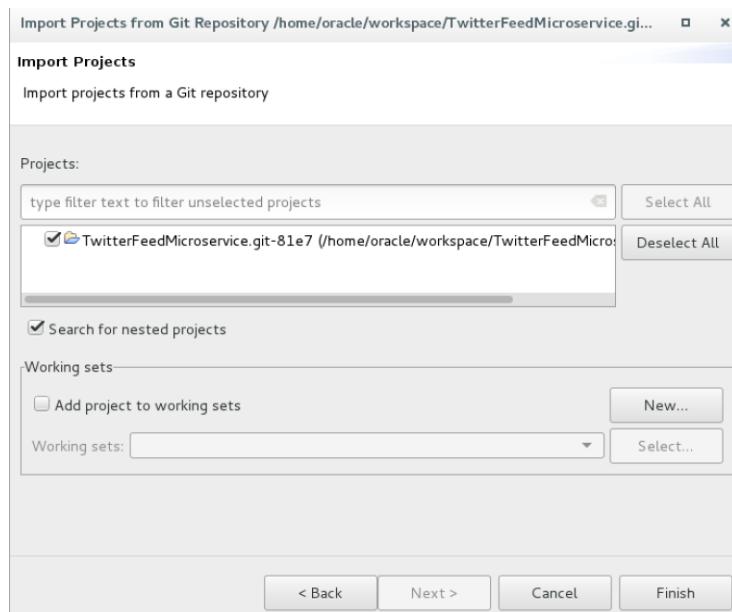
- Right Click** on the **TwitterFeedMicroservice** cloned repository and **click** on **Import Projects**.



- Keep the wizard defaults and **click on Next**

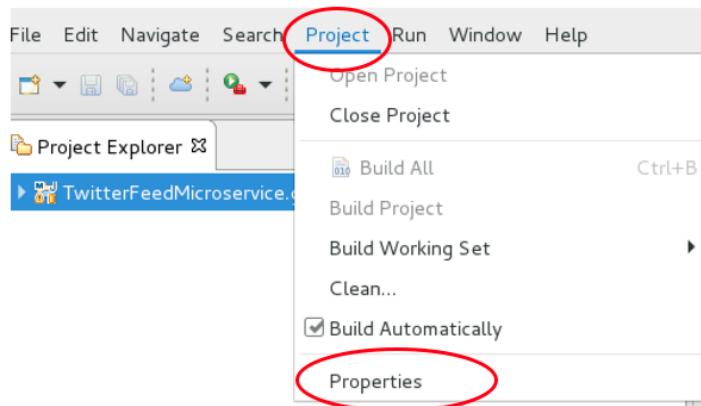


- Accept the Import defaults, and **click on Finish**

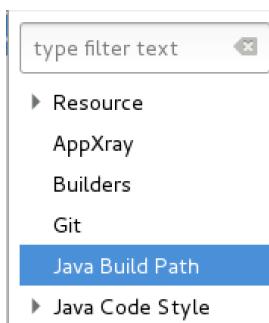


STEP 15: Select the correct Java JDE

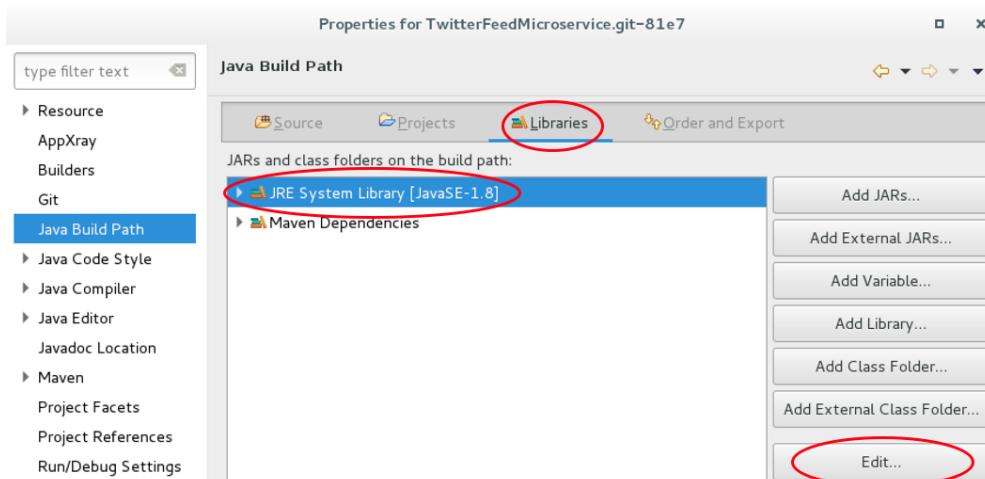
- Click on the **TwitterFeedMicroservice** Project, then from the top menu, select **Project > Properties**



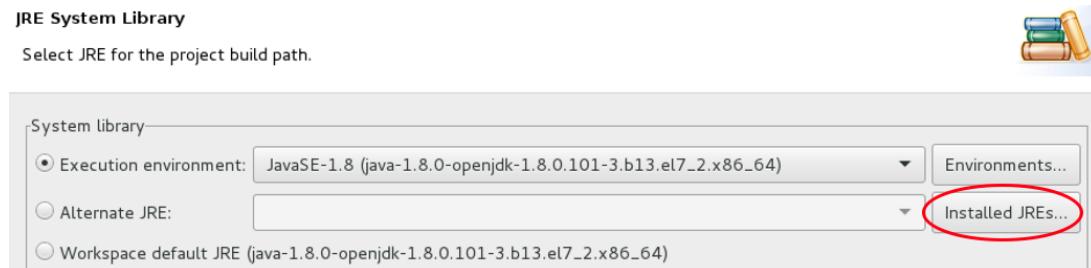
- Select the **Java Build Path** option.



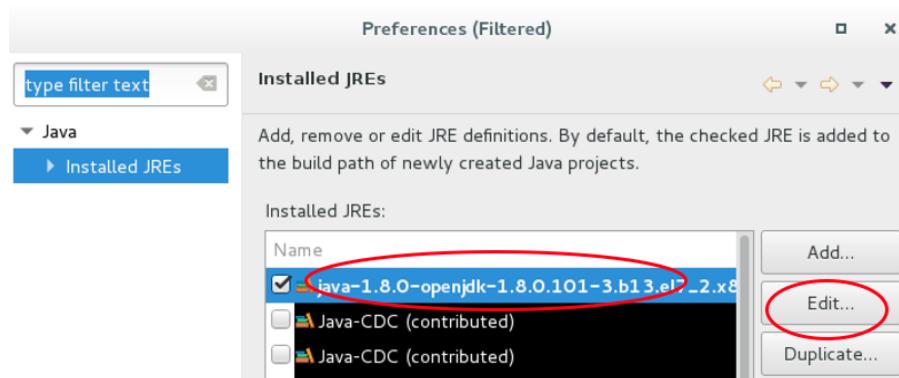
- Click on the **Libraries** tab, then select the **JRE System Library**. Next, click on the **Edit** button.



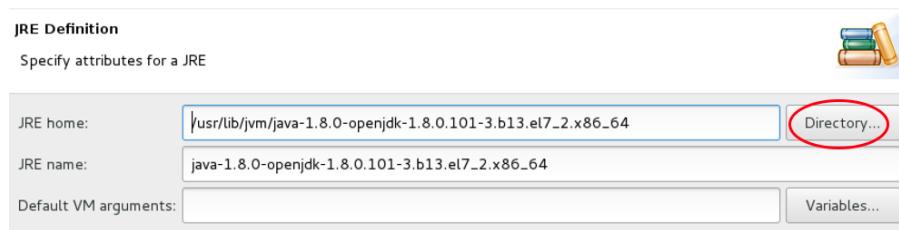
- Click** on the **Installed JREs** button.



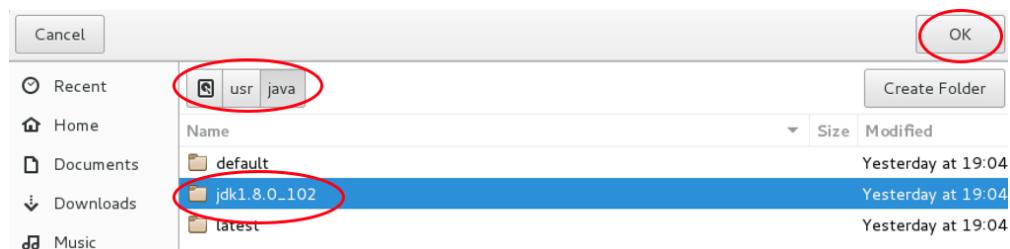
- Select** the Standard VM, which in this case is **java-1.8.0-openjdk**. Then, **click** on **Edit**



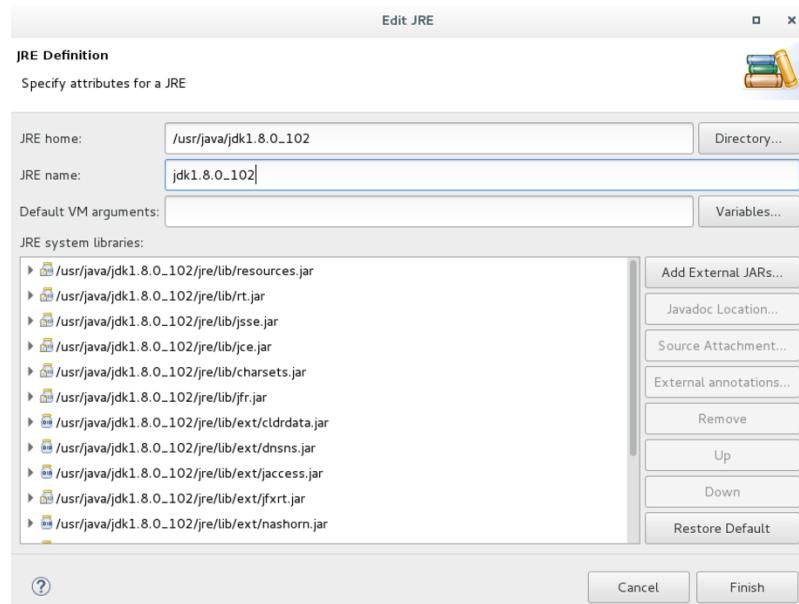
- Click** on the **JRE home: Directory** button



- Navigate** to **usr/java**, select **jdk1.8.0_102**, and **click** on **OK**



- Change the JRE Name to **jdk1.8.0_102**, and click on **Finish**



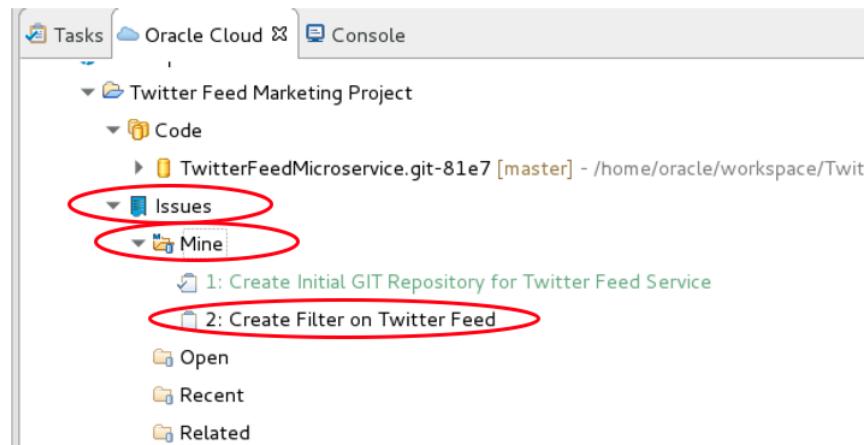
- Click **OK**, **Finish** and **OK** when prompted on the following dialogs to complete the Library changes.

Test the Local Cloned Services

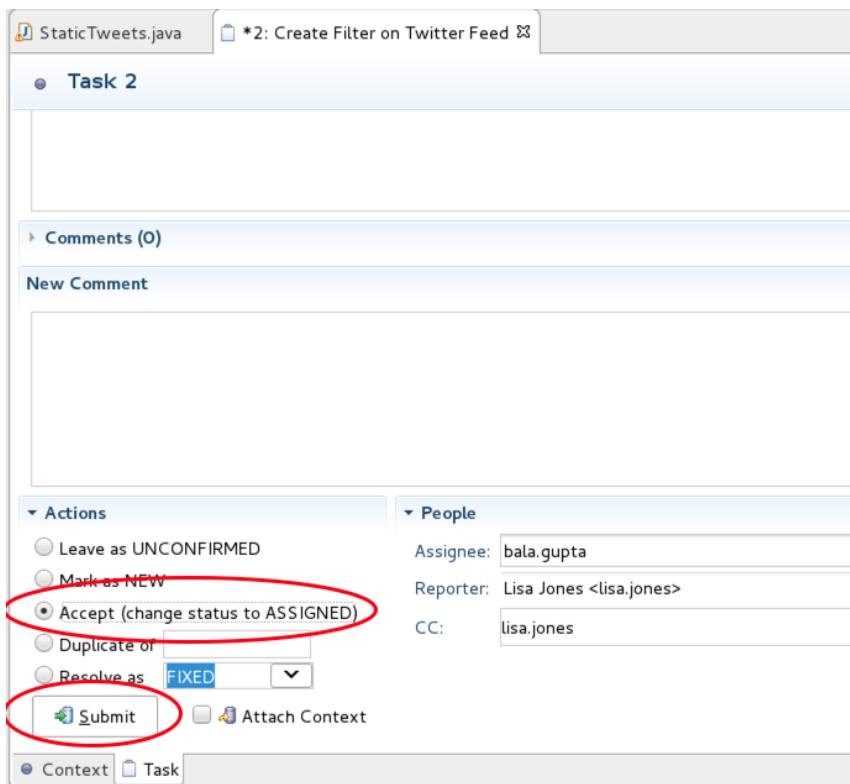
STEP 16: Set Feature 2 Status to In Progress

In the previous steps we updated the status of the Tasks assign to Bala Gupta using the web interface to the Developer Cloud Service. In this step we will use the Eclipse connection to the Developer Cloud Service to update the status of Bala's tasks.

- Within the Oracle Cloud Connection tab, double click the **Issues** to expand, then double click on **Mine** to expand your list. Once you see the list of your Issues, then double click on **Create Filter on Twitter Feed**.



- Scroll down to the bottom of the **Create Filter on Twitter Feed** window. In the **Actions section**, and change the Actions to **Accept (change status to ASSIGNED)**, then click on **Submit**.

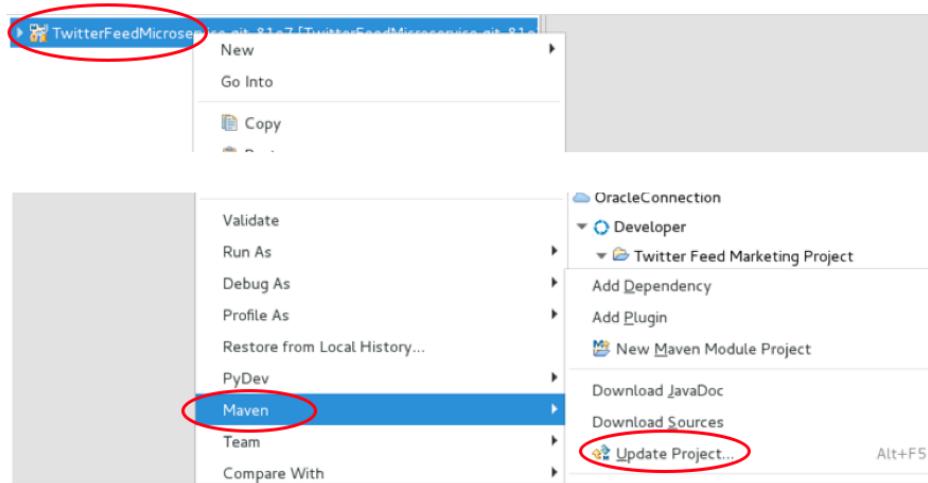


- Optionally, if you return to the Developer Cloud Service web interface, you'll see that the Eclipse interface caused the Feature 2 to be moved to the "In Progress" column of the Agile > Active Sprints.

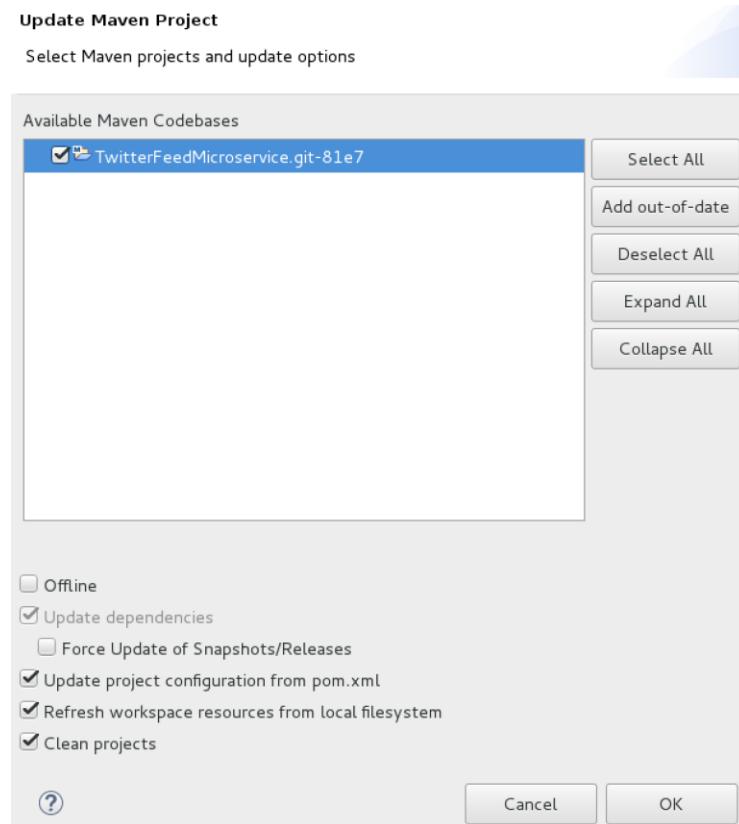
The screenshot shows the Oracle Public Cloud Workshop web interface. At the top, there's a 'Microservices' dropdown and a 'Sprint' filter set to 'Sprint 1 – Initial Development -- 11/2/2016 1:13 PM • 11/16/2016 12:13 PM'. A 'Complete Sprint' button is also visible. Below this, there are filters for 'Sort Issues by: Priority' and 'Sort Swimlanes by: User Name'. The main board displays two columns: 'To Do (2)' and 'In Progress (1)'. The 'To Do' column has 3 items. The 'In Progress' column has 1 item, labeled 'Feature 2: Create Filter on Twitter Feed', assigned to 'bala.gupta'. The status of this item is 'ASSIGNED'.

STEP 17: Build and test the TwitterFeedMicroservice

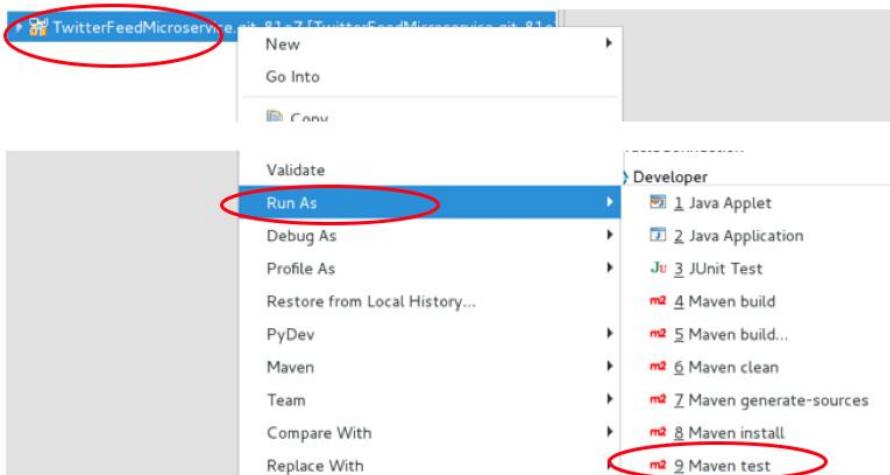
- Right Click** on the **TwitterFeedMicroservice** project. Select **Maven > Update Project**



- Keep the defaults, and click **OK**. This will run Maven, and build the project.



- To test the local copy of the project code, right click on the **TwitterFeedMicroservice** project and select **Run As > Maven Test**



- Double Clicking on the **Console tab** will expand The Window. You can minimize the Window by double clicking the tab again. If the TwitterFeedMicroservices test runs successfully, your console window will contain the following – Notice the message with “### Tweets in Static Tweets”. Also, you should see that there were zero Failures.

<terminated> /usr/java/jdk1.8.0_102/bin/java (Aug 17, 2016, 4:15:37 PM)
[INFO] Downloaded: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/surefire-

T E S T S

Running com.example.MyServiceTest
Aug 17, 2016 4:15:52 PM org.glassfish.grizzly.http.server.NetworkListener start
INFO: Started listener bound to [localhost.localdomain:8080]
Aug 17, 2016 4:15:52 PM org.glassfish.grizzly.http.server.HttpServer start
INFO: [HttpServer] Started
{"delete": {"status": {"id": "765578381903601664", "id_str": "765578381903601664", "user_id": 2},
"delete": {"status": {"id": "765625802700513281", "id_str": "765625802700513281", "user_id": 7},
"delete": {"status": {"id": "519997185069305856", "id_str": "519997185069305856", "user_id": 1},
"created_at": "Tue Aug 16 19:09:09 +0000 2016", "id": "765626469619859456", "id_str": "765626469619859456",
"created_at": "Tue Aug 16 19:09:09 +0000 2016", "id": "765626469615702016", "id_str": "765626469615702016",
"created_at": "Tue Aug 16 19:09:09 +0000 2016", "id": "765626469582110721", "id_str": "765626469582110721",
"created_at": "Tue Aug 16 19:09:09 +0000 2016", "id": "765626469607354368", "id_str": "765626469607354368",
"created_at": "Tue Aug 16 19:09:09 +0000 2016", "id": "76562646956158865856", "id_str": "76562646956158865856",
"created_at": "Tue Aug 16 19:09:09 +0000 2016", "id": "765626469586497536", "id_str": "765626469586497536",
"created_at": "Tue Aug 16 19:09:09 +0000 2016", "id": "765626469590634497", "id_str": "765626469590634497"},
101 Tweets in StaticTweets
Aug 17, 2016 4:15:53 PM org.glassfish.grizzly.http.server.NetworkListener shutdownNow
INFO: Stopped listener bound to [localhost.localdomain:8080]
Aug 17, 2016 4:15:53 PM org.glassfish.grizzly.http.server.NetworkListener start
INFO: Started listener bound to [localhost.localdomain:8080]
Aug 17, 2016 4:15:53 PM org.glassfish.grizzly.http.server.HttpServer start
INFO: [HttpServer-1] Started.
Aug 17, 2016 4:15:53 PM org.glassfish.grizzly.http.server.NetworkListener shutdownNow
INFO: Stopped listener bound to [localhost.localdomain:8080]
Aug 17, 2016 4:15:54 PM org.glassfish.grizzly.http.server.NetworkListener start
INFO: Started listener bound to [localhost.localdomain:8080]
Aug 17, 2016 4:15:54 PM org.glassfish.grizzly.http.server.HttpServer start
INFO: [HttpServer-2] Started.
Tests run: 3, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 3.593 sec
Aug 17, 2016 4:15:54 PM org.glassfish.grizzly.http.server.NetworkListener shutdownNow
INFO: Stopped listener bound to [localhost.localdomain:8080]

Results :

Tests run: 3, Failures: 0, Errors: 0, Skipped: 0

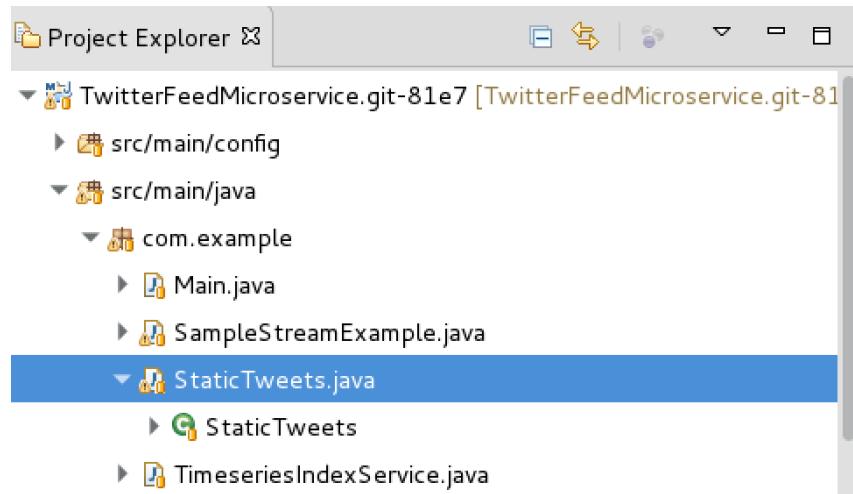
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 14.560 s
[INFO] Finished at: 2016-08-17T16:15:54-04:00
[INFO] Final Memory: 11M/46M
[INFO] -----

Add the Filter to the Service

The Code we cloned locally contains all the source necessary to filter the Static Twitter Feed. In this section of the lab, we will un-comment the code and test the filter.

STEP 18: Add Filter

- In the Project Explorer, expand the **TwitterFeedMicroservice > src/main/java > com.example** and double click on **StaticTweets.java** to open the source code.



- In the StaticTweets.java source file, scroll down until you find two lines of code that begin with "**--- Remove this comment**". **Delete** both of **these lines** to activate the code that will cause filtering of the Static Tweets file to occur.

```
/**
 * Method handling HTTP GET requests. The returned object will be sent
 * to the client as "text/json" media type.
 *
 * @return String that will be returned as an application/json response.
 */
/* --- Remove this comment to allow for Filtered Searching */
@Path("{search}")
@GET
@Produces(MediaType.APPLICATION_JSON)
public Response getItWithCount(@PathParam("search") final String search) {
    final ChunkedOutput<String> output = new ChunkedOutput<String>(String.class);
    runTask(output, search);
    return Response.ok()
        .entity(output)
        .header("Access-Control-Allow-Origin", "*")
        .header("Access-Control-Allow-Methods", "GET, POST, DELETE, PUT")
        .build();
}

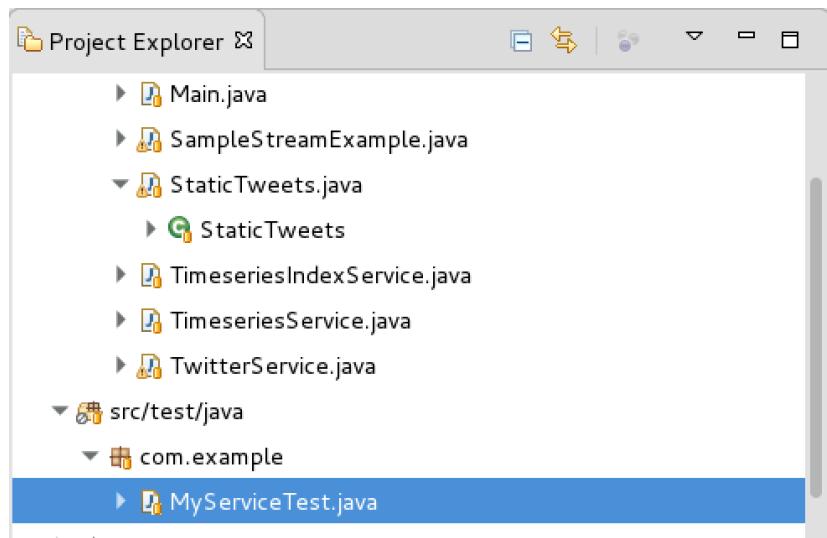
/* --- Remove this comment to allow for Filtered Searching */
```

- Your code should now look like this:

```
  /**
   * Method handling HTTP GET requests. The returned object will be sent
   * to the client as "text/json" media type.
   *
   * @return String that will be returned as an application/json response.
   */
  @Path("{search}")
  @GET
  @Produces(MediaType.APPLICATION_JSON)
  public Response getItWithCount(@PathParam("search") final String search) {
    final ChunkedOutput<String> output = new ChunkedOutput<String>(String.class);
    runTask(output, search);
    return Response.ok()
      .entity(output)
      .header("Access-Control-Allow-Origin", "*")
      .header("Access-Control-Allow-Methods", "GET, POST, DELETE, PUT")
      .build();

}
```

- Next we will enable the filter in the testing code. Expand the **src/test/java > com.example** folder, and double click on **MyServiceTest.java** to open the source file



- In the MyServiceTest.java source file, locate the method **testGetStaticSearchTweets()**, and **remove** the **comments** so that section of code will execute.

The screenshot shows an IDE interface with three tabs at the top: StaticTweets.java, *MyServiceTest.java (which is the active tab), and SampleStreamExample.java. The code in *MyServiceTest.java is as follows:

```
    @After
    public void tearDown() throws Exception {
        server.shutdown();
    }

    /**
     * Test to see that the message "Got it!" is sent in the response.
     */

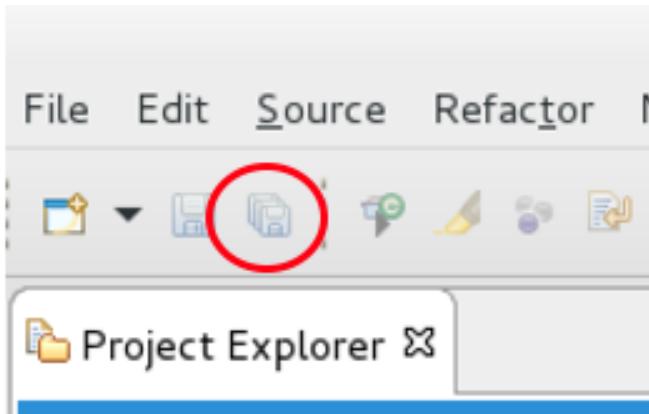
    @Test
    public void testGetStaticTweets() {
        String responseMsg = target.path("statictweets").request().get(
            assertNotNull(responseMsg);
    }

    /*-----*/
    @Test
    public void testGetStaticSearchTweets() {
        String responseMsg = target.path("statictweets/alpha").request(
            assertNotNull(responseMsg);
    }

    /*-----*/
    /*
     * Test
     */
    @Test
    public void testGetTweets() {
        String responseMsg = target.path("tweets").request().get(String
            assertNotNull(responseMsg);
    }
}
```

Two sections of code are highlighted with red dashed boxes: the `testGetStaticSearchTweets()` method and the `testGetTweets()` method. The `testGetTweets()` method is also preceded by a multi-line comment block.

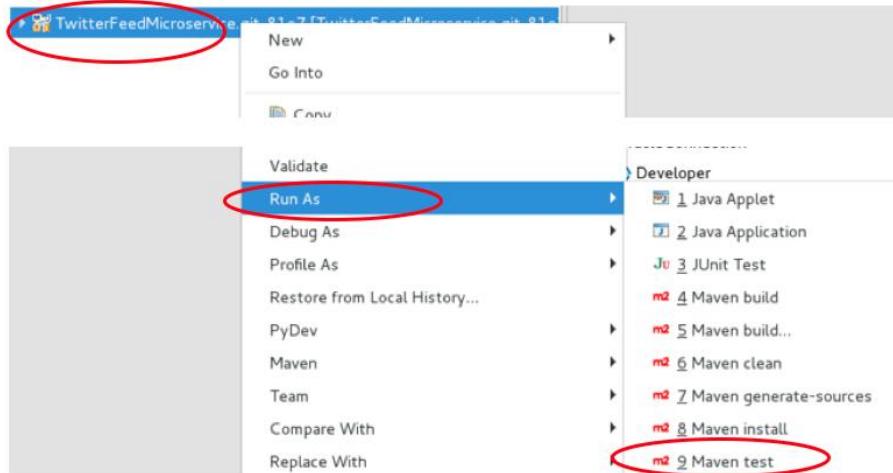
- Click on the **Save All** icon



Test the Local Filtered Services

STEP 19: Run Test

- Run the Test by right clicking on **TwitterFeedMicroservice** and selecting **Run As > Maven Test**



- Once the test runs, you'll see the Static Twitter feed returned for both the unfiltered and filtered tests. You should not see any Failures.

```

<terminated> /usr/java/jdk1.8.0_102/bin/java (Aug 17, 2016, 5:31:07 PM)
{"delete": {"status": {"id": 765578381903601664, "user_id": 2186468947, "user_id_str": "2186468947", "text": "The V. #Dolci @Cody"}, "id": 765625802700513281, "user_id": 72625334, "user_id_str": "72625334", "text": "RT @sl: #kidr"}, {"delete": {"status": {"id": 16693090, "user_id": 16693090, "user_id_str": "16693090", "text": "I've 1 2 Tweets in StaticTweets"}, "id": 519997185069305856, "user_id": 16693090, "user_id_str": "16693090", "text": "RT @sl: #kidr"}, {"created_at": "Tue Aug 16 19:09:00 +0000 2016", "id": 765626469619859456, "id_str": "765626469619859456", "text": "The V. #Dolci @Cody"}, {"created_at": "Tue Aug 16 19:09:00 +0000 2016", "id": 765626469615702016, "id_str": "765626469615702016", "text": "RT @sl: #kidr"}, {"created_at": "Tue Aug 16 19:09:00 +0000 2016", "id": 765626469582110721, "id_str": "765626469582110721", "text": "@Cody RT @sl: #kidr"}, {"created_at": "Tue Aug 16 19:09:00 +0000 2016", "id": 765626469607354368, "id_str": "765626469607354368", "text": "RT @sl: #kidr"}, {"created_at": "Tue Aug 16 19:09:00 +0000 2016", "id": 765626469615865856, "id_str": "765626469615865856", "text": "@Cody RT @sl: #kidr"}, {"created_at": "Tue Aug 16 19:09:00 +0000 2016", "id": 765626469586497536, "id_str": "765626469586497536", "text": "RT @sl: #kidr"}, {"created_at": "Tue Aug 16 19:09:00 +0000 2016", "id": 765626469590634497, "id_str": "765626469590634497", "text": "@Cody RT @sl: #kidr"}}

Aug 17, 2016 5:31:16 PM org.glassfish.grizzly.http.server.NetworkListener shutdownNow
INFO: Stopped listener bound to [localhost.localdomain:8080]
Aug 17, 2016 5:31:16 PM org.glassfish.grizzly.http.server.NetworkListener start
INFO: Started listener bound to [localhost.localdomain:8080]
Aug 17, 2016 5:31:16 PM org.glassfish.grizzly.http.server.HttpServer start
INFO: [HttpServer-1] Started.
{"created_at": "Tue Aug 16 19:09:00 +0000 2016", "id": 765626469619859456, "id_str": "765626469619859456", "text": "Alpha"}, {"created_at": "Tue Aug 16 19:09:10 +0000 2016", "id": 765626473797545984, "id_str": "765626473797545984", "text": "I've 1 2 Tweets in StaticTweets"}, {"created_at": "Tue Aug 16 19:09:10 +0000 2016", "id": 765626469615865856, "id_str": "765626469615865856", "text": "@Cody RT @sl: #kidr"}, {"created_at": "Tue Aug 16 19:09:10 +0000 2016", "id": 765626469586497536, "id_str": "765626469586497536", "text": "RT @sl: #kidr"}, {"created_at": "Tue Aug 16 19:09:10 +0000 2016", "id": 765626469590634497, "id_str": "765626469590634497", "text": "@Cody RT @sl: #kidr"}}

Aug 17, 2016 5:31:17 PM org.glassfish.grizzly.http.server.NetworkListener shutdownNow
INFO: Stopped listener bound to [localhost.localdomain:8080]
Aug 17, 2016 5:31:17 PM org.glassfish.grizzly.http.server.NetworkListener start
INFO: Started listener bound to [localhost.localdomain:8080]
Aug 17, 2016 5:31:17 PM org.glassfish.grizzly.http.server.HttpServer start
INFO: [HttpServer-2] Started.
Aug 17, 2016 5:31:17 PM org.glassfish.grizzly.http.server.NetworkListener shutdownNow
INFO: Stopped listener bound to [localhost.localdomain:8080]
Aug 17, 2016 5:31:17 PM org.glassfish.grizzly.http.server.NetworkListener start
INFO: Started listener bound to [localhost.localdomain:8080]
Aug 17, 2016 5:31:17 PM org.glassfish.grizzly.http.server.HttpServer start
INFO: [HttpServer-3] Started.

Tests run: 4, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 4.536 sec
Aug 17, 2016 5:31:17 PM org.glassfish.grizzly.http.server.NetworkListener shutdownNow
INFO: Stopped listener bound to [localhost.localdomain:8080]

Results:

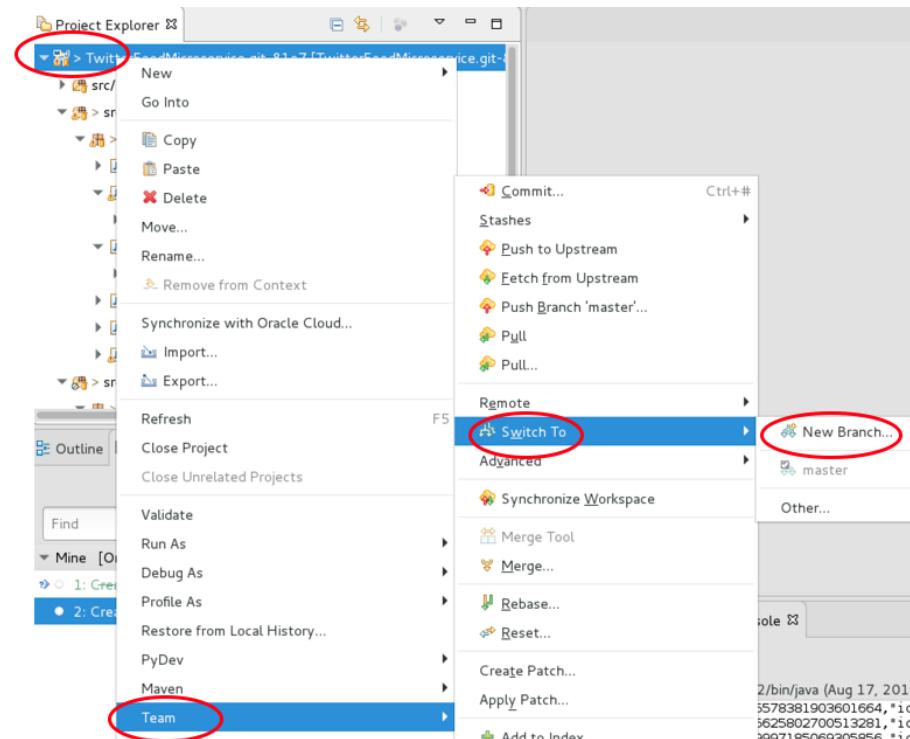
Tests run: 4, Failures: 0, Errors: 0, Skipped: 0
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
```

Create a new Branch and Commit Code

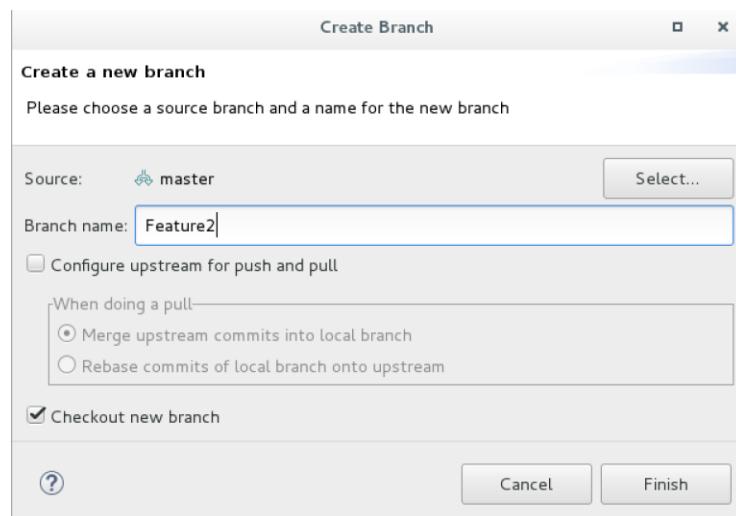
Create a Branch and Commit Code

STEP 20: Create a new Branch and Commit Code

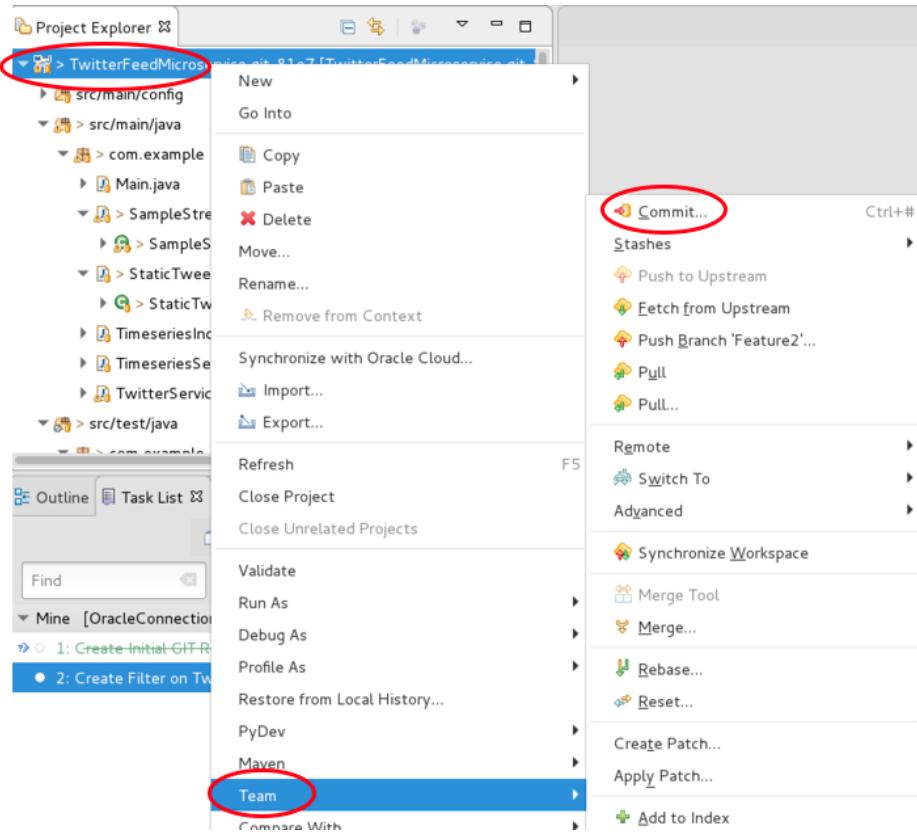
- To create a new branch in the Git repository, right click on **TwitterMicroservice** and then Select **Team > Switch To > New Branch**



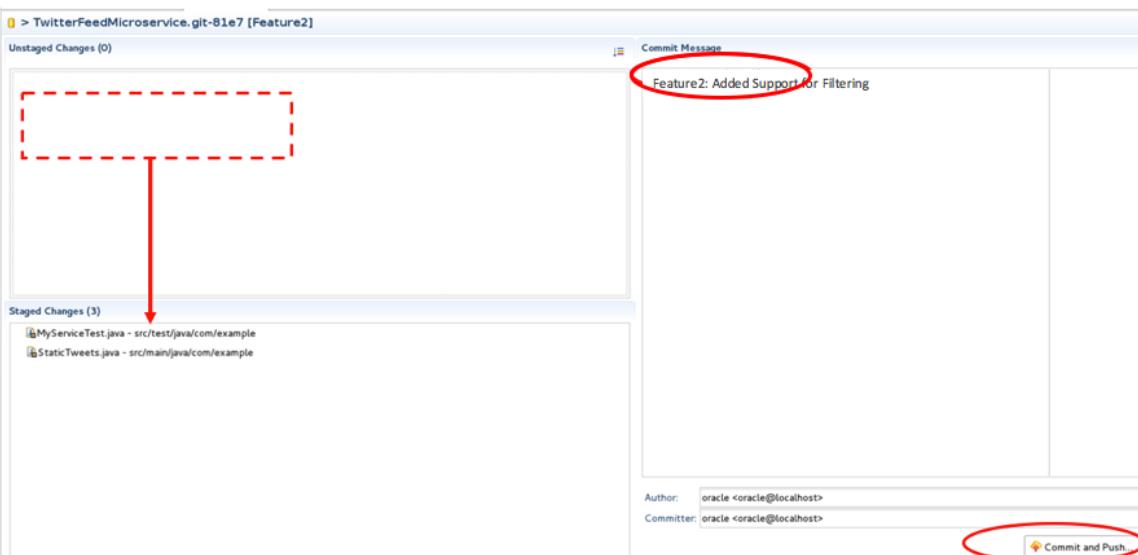
- Enter **Feature2** for the Branch name, and click on **Finish**



- We can now commit our code to the branch by Right Clicking on **TwitterFeedMicroservice** and then selecting **Team > Commit**



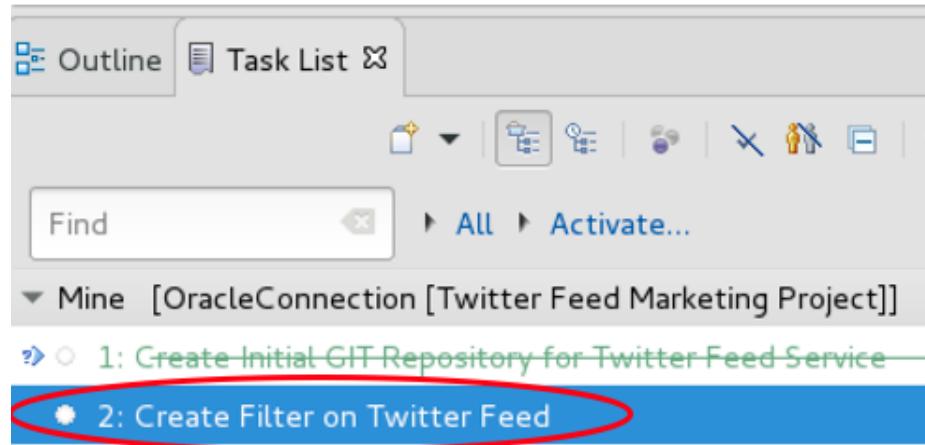
- Enter “**Feature2: Added Support for Filtering**” in the Commit Message box, **Drag and Drop the changed files** into the **Staged Changes** panel, and click on **Commit and Push**. Note: it is possible to change the default Author and Committer to match the current “persona.” However, for the sake of this lab guide, we will leave the defaults.



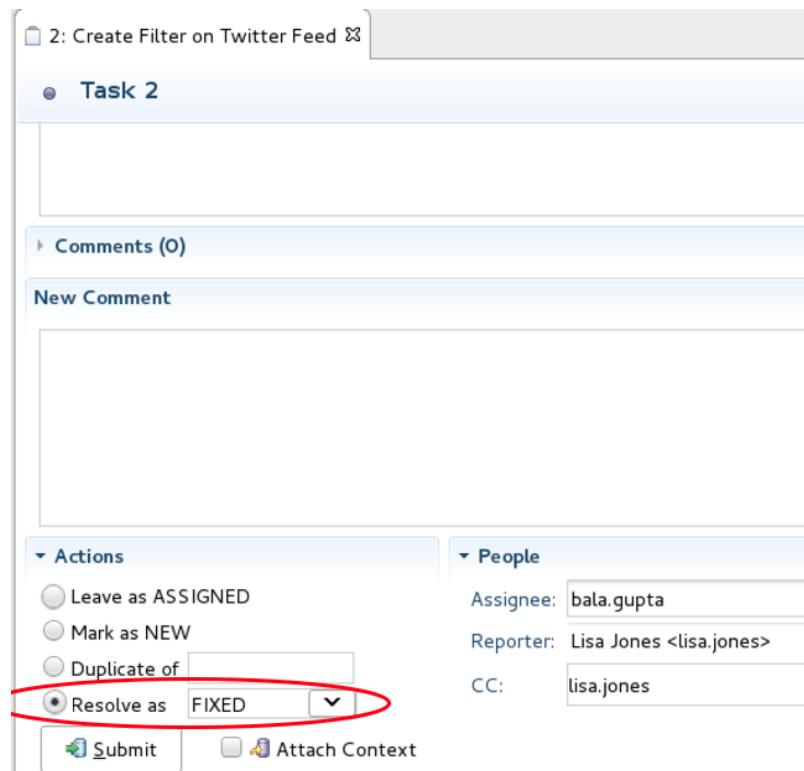
- Accept the Default for the **Push Branch Feature 2** dialog and click on **Next**
- Click on the **Finish** button in the Push Confirmation dialog
- Click on **Ok** in Push Result dialog

STEP 21: Complete the Create Filter Task

- In the lower left Eclipse Task List, double click on **Create Filter on Twitter Feed** task



- In the **Create Filter on Twitter Feed** window, scroll down to the **Actions** Section. Click on **Resolve as FIXED**, and then click on the **Submit** button



Create Merge Request

STEP 22: Review Sprint Status and create Merge Request

- Return to the Developer Cloud Service Dashboard in the browser, and select **Agile**. If your default Board is not set to Microservices, then set the Find Board Filter to All, and select the Microservices board.
- Click on **Active Sprints** button. Notice that **Feature 2** is now in the **Verify Code** column

The screenshot shows the Agile board with two items in the 'Verify Code' column:

- Feature 2:** Create Filter on Twitter Feed. Status: RESOLVED : FIXED. Points: 2.
- Task 1:** Create Initial GIT Repository for Twitter Feed Service. Status: VERIFIED - FIXED. Points: 1.

- Next, on navigation panel click **Code**, select the **Feature2** branch and then click on the **Commits** sub tab. Now view the recent commit made to branch from within Eclipse.

The screenshot shows the Code interface for the Twitter Feed Marketing Project. The Feature2 branch is selected. The 'Commits' tab is highlighted with a red circle.

- Now that Bala Gupta has completed the task of adding the search filter, a Merge Request can be created by Bala and assigned to Lisa Jones. Click on the **Merge Requests**, and then click on the **New Request button**.

The screenshot shows the Merge Requests interface. The 'New Merge Request' button is highlighted with a red circle.

- Enter the following information into the New Merge Request and click **Next**

Repository: TwitterFeedMicroservice.git
 Target Branch: master
 Review Branch: Feature2

New Merge Request X

[Back](#) Branch Details Description [Next >](#)

*	Repository	TwitterFeedMicroservice.git	▼
*	Target Branch	master	▼
*	Review Branch	Feature2	▼

oracle Today
 3b75f2a Feature2: Added Support for Filtering

- Enter the following information into **Details** and click **Create**

Summary: Merge Feature 2 into master
 Reviewers: Lisa Jones (or current user in non-multi user env)

New Merge Request X

[Back](#) Branch Details Description [Next >](#)

Linked Issues	Search and Link Issues
Linked Builds	Search and Link Build Jobs
Summary	Merge Feature 2 into master
* Reviewers	Lisa Jones X

- In the **Write** box, enter the following comment and then click on the **Comment** button to save: "I added the ability to add a filter request to the end of the URL - e.g. statictweets/alpha"

The screenshot shows a GitHub commit page. At the top, it displays a profile icon and the text "Bala Gupta added 1 commits 28 minutes ago". Below this is a list of commits, with one commit shown: "4a28ae0 Added Support for Filtering". The main area has two tabs: "Write" (which is selected) and "Preview". In the "Write" tab, there is a text input field containing the text "I added the ability to add a filter request to the end of the URL - i.e. statictweets/alpha". To the right of the input field is a "Markdown Reference" link. At the bottom right of the "Write" tab is a "Comment" button.

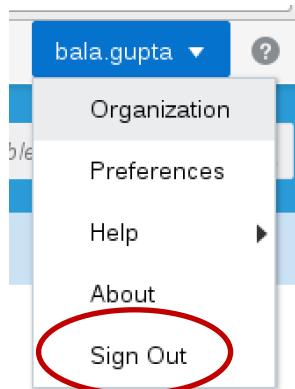
Merge the Branch as Lisa Jones

In the following steps "Lisa" will merge the branch created by "Bala" into the master.

NOTE: If you are using a single user environment, you will skip the next step, and go to the following step titled: "Merge Requests"

STEP 23: Sign Out as Bala Gupta and Sign In as Lisa Jones

- Click on the **bala.gupta** dropdown located in the top right corner of the screen. Select **Sign Out**.



- Following the previously documented steps, go to the URL: <http://cloud.oracle.com>, click on **Sign In** found on the Top Right corner of the window. Select the correct **Data Center**, click on the **My Services** button, enter the correct **Identity Domain** and click on **Go**.
- Enter **lisa.jones** for the username, and enter the correct password. Click on **Sign In**.

Welcome gse00002057 [change domain](#) [?](#)

lisa.jones

••••••••••••••

[Can't access your account?](#)

Sign In

- When the Dashboard is displayed, click on the **Developer Cloud Service**. Note: It is possible that based on your browser cookie settings, you may be automatically taken to the Developer screen at the end of this step.

≡ ORACLE CLOUD My Services

Account: gse00002057 ▾

0 Important Notifications

Create Instance ▾ Customize Dashboard ▾

Dashboard

Switch to original Dashboard

developer71725	Database	Compute	Database Backup
1 Instances	1 Instances	1 Instances	
Storage	Java	Application Container	SOA
0 Instances	0 Instances	0 Instances	0 Instances

- From the Developer Cloud Service Dashboard, click on the **Open Service Console** button

Service Details: developer71725 (Oracle Developer Cloud Service)

Service Status - August 2016

Month View | Quarterly View | Year View | Current Month

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

Legend: Before Activation (Grey) Service Up (Green) Planned Outage (Yellow) Service Incident (Red)

Overview
(for August 2016)
100% uptime

Business Metrics
(as of 10 hours 53 minutes ago)
0 current disk usage

Additional Information

Plan: Trial Developer Service
Service Start Date: 26. Juli 2016
Service End Date: Not available
Subscription ID: 536840104
Customer Account: gse00002057 (US)
CSI Number: Not available

Data Center: EMEA Commercial 2 - Amsterdam
Version: 16.1.0.0
Status: Active
Service Instance URL: <https://developer.em2.oracle.com>

- Select the **Twitter Feed Marketing Project**

Member Favorites Owner All

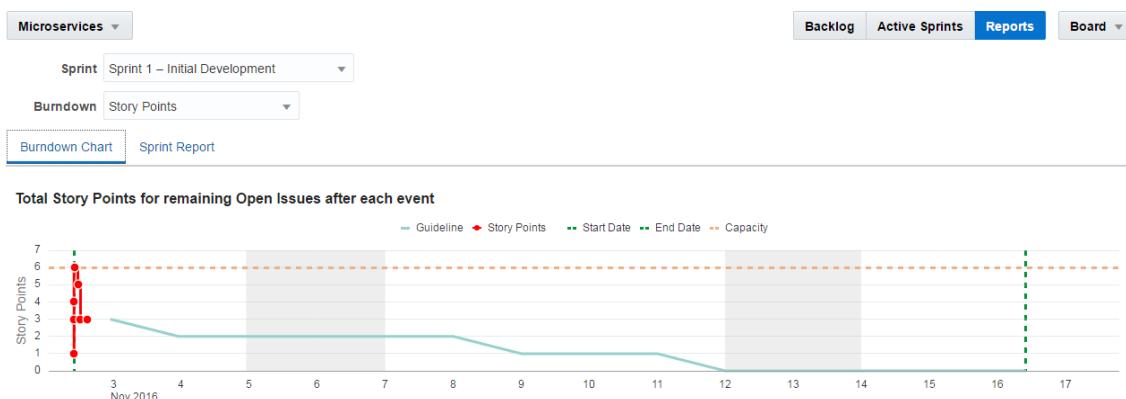
New Project

Filter Projects

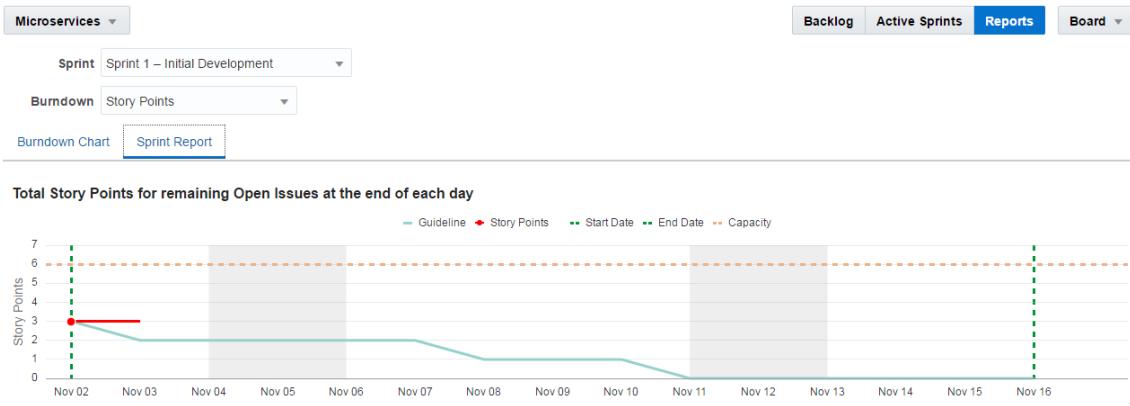
Twitter Feed Marketing Project
Project to gather and analyze twitter data

STEP 24: Merge Requests

- Before moving forward, "Lisa Jones" can take a look at the **Burndown** and **Sprint Reports** by clicking on the **Agile** navigation, then the **Reports** button



Click **Sprint Report**



- On navigation panel click **Merge Requests**. Select the **Assigned to Me** search. After the search completes, click on the **Merge Feature 2 into master** assigned request.

Project

Code

Maven

Snippets

Merge Requests

Issues

Agile

Build

ORACLE® Developer Cloud Service

Twitter Feed Marketing Project

Merge Requests

Standard Searches Assigned To Me

Created By Me

Assigned To Me

All Open Requests

All Closed Requests

All Requests

ID	Summary	Status
5	Merge Feature 2 into master	OPEN

Page 1 of 1 (1 of 1 items) < 1 > >>

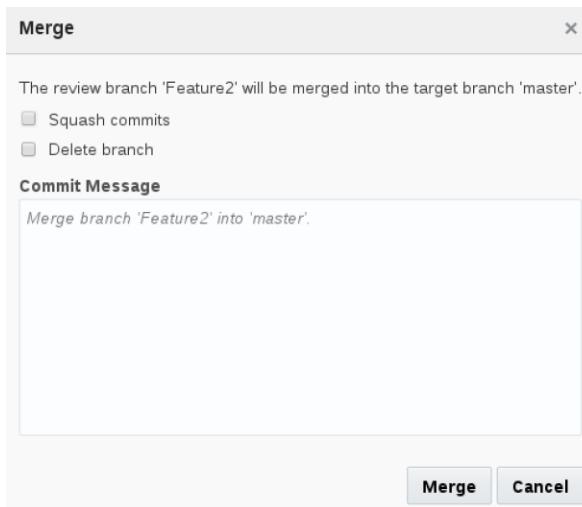
- Once the request has loaded, select the **Changed Files** tab. "Lisa" will now have the opportunity to review the changes in the branch, make comments, request more information, etc. before Approving, Rejecting or Merging the Branch.

The screenshot shows a GitHub pull request interface. At the top, it says "Merge Feature 2 into master". Below that, it says "OPEN Bala Gupta wants to merge 1+ commits to master from Feature2". There are four tabs at the top: "Conversation", "Commits (1+)", "Changed Files (3)" (which is selected and highlighted in blue), and "Linked Issues". The main area shows the content of a file named "StaticTweets.java" with line numbers 45 to 65. Several lines of code are highlighted in red, specifically lines 48, 63, and 64, with the message "/* --- Remove this comment to allow for Filtered Searching */" appearing twice. The code itself includes annotations like @Path("{search}") and @GET.

- Click on the **Merge** button.



- Leave the defaults, and click on the **Merge** button in the confirmation dialog.



- Now that the code has been committed to the Developer Cloud Service repository, the build and deployment will automatically start. On the navigation panel click **Build**, and you should see a **Twitter Feed Build** in the Queue

Jobs Overview

Build Queue Twitter Feed Build 42%

[View Build History](#)

+ New Job **All**

Status	Weather	Job	Last Success
		Twitter Feed Build	Today at 12:11 PM -0600

- Wait a minute or two for the build to complete. The **Last Success** will be set to **Just Now** when the build completes.

Jobs Overview

Build Queue

View Build History

New Job All

Status	Weather	Job	Last Success
		Twitter Feed Build	Just now

- Click **Deploy**. Wait for the Deploy Status to change to **Deployment update in progress**, and then change to **Last deployment succeeded – Just now**.

Deployments

JavaTwitterMicroservice

Deploy to ACCS em2 / gse00002921 / bala.gupta
 Configuration TwitterFeedMicroserviceDeploy
 Job / Build Twitter Feed Build / Latest Successful Build
 Artifact target/twitter-microservice-example-dist.zip

Deployment update in progress

Deployments

JavaTwitterMicroservice

Deploy to ACCS em2 / gse00002921 / bala.gupta
 Configuration TwitterFeedMicroserviceDeploy
 Job / Build Twitter Feed Build / Latest Successful Build
 Artifact target/twitter-microservice-example-dist.zip

Last deployment succeeded -- Just now.

Test the JavaTwitterMicroservice in the Cloud

- Once the service has successfully deployed, click on the [JavaTwitterMicroservice](#) link

Deployments

Deploy to ACCS em2 / gse00002921 / bala.gupta
 Configuration TwitterFeedMicroserviceDeploy
 Job / Build Twitter Feed Build / Latest Successful Build
 Artifact target\twitter-microservice-example-dist.zip
 ✓ Last deployment succeeded -- Just now.

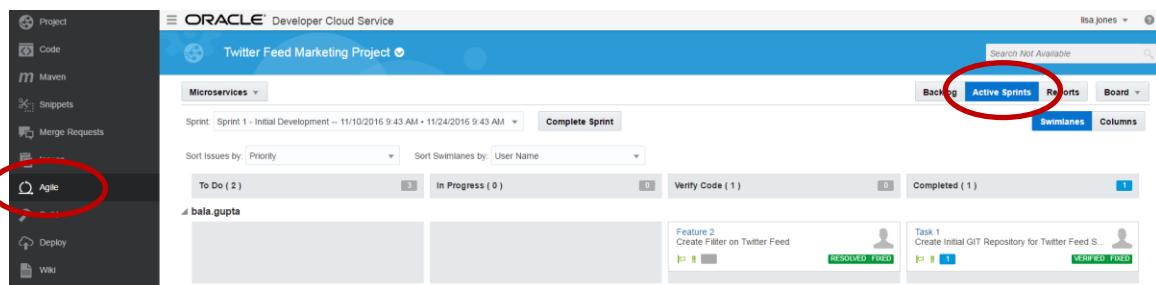
- When the new browser tab loads, Append [/statictweets](#) to the end of the URL and **press enter** to test the original static twitter service

```
{"tweets": [{"delete": {"status": {"id": 765578381903601664, "id_str": "765578381903601664", "user": {"id": 765625802700513281, "id_str": "765625802700513281", "user_id": 72625334, "user_id_str": "72625334", "name": "Nick Courtney", "screen_name": "NickCourtney", "location": "http://www.book.events and Sweet Memories Vinyl Records", "url": "https://www.facebook.com/nickcourtney", "protected": false, "verified": false, "followers_count": 2009, "utc_offset": 3600, "time_zone": "London", "geo_enabled": false, "lang": "en", "contributors_enabled": false, "profile_background_image_url_https": "https://abs.twimg.com/images/themes/theme1/bg.gif", "profile_text_color": "#EFEFEF", "profile_use_background_image": true, "profile_image_url": "https://pbs.twimg.com/profile_images/697776971057594368/kMfUzOxw_normal.jpg", "profile_banner_url": "https://pbs.twimg.com/profile_banners/697776971057594368/1474544000", "geo": null, "coordinates": null, "place": null, "contributors": null}, "is_quote_status": false, "text": "#DolceAmoreHotSeat Wave 2016", "id": 765626469615702016, "id_str": "765626469615702016", "truncated": false, "in_reply_to_status_id": null, "in_reply_to_user_id": null, "in_reply_to_screen_name": null, "retweet_count": 0, "favorite_count": 0, "entities": {"urls": [{"url": "https://t.co/9DGpEX0js", "expanded_url": "http://fb.me/7y5d5kFTD", "display_url": "fb.me/7y5d5kFTD"}, {"url": "https://t.co/9DGpEX0js", "expanded_url": "http://t.co/9DGpEX0js", "display_url": "t.co/9DGpEX0js"}], "hashtags": [{"text": "#DolceAmoreHotSeat"}], "symbols": [{"text": "#"}]}, "favorited": false, "retweeted": false, "possibly_sensitive": false, "filter_level": "low", "lang": "en"}, {"delete": {"status": {"id": 724990427187671041, "id_str": "724990427187671041", "user": {"id": 16693090, "id_str": "16693090", "user_id": 16693090, "user_id_str": "16693090", "name": "LizQuenlurve", "screen_name": "Philippines", "location": "Philippines", "url": null, "description": null, "protected": false, "verified": false, "followers_count": 0, "friends_count": 0, "listed_count": 0, "statuses_count": 0, "geo_enabled": false, "lang": "en", "contributors_enabled": false, "profile_background_tile": false, "profile_link_color": "#2B7BB9", "profile_sidebar_border_color": "#2B7BB9", "profile_sidebar_fill_color": "#2B7BB9", "profile_text_color": "#2B7BB9", "profile_use_background_image": true, "profile_image_url": "https://pbs.twimg.com/profile_images/1000000000000000000/placeholderimage.png", "profile_banner_url": "https://pbs.twimg.com/profile_banners/16693090/1474544000"}, "is_quote_status": false, "text": "TweetDeck", "id": 724990427187671041, "id_str": "724990427187671041", "truncated": false, "in_reply_to_status_id": null, "in_reply_to_user_id": null, "in_reply_to_screen_name": null, "retweet_count": 0, "favorite_count": 0, "entities": {"urls": [{"url": "http://fb.me/7y5d5kFTD", "expanded_url": "http://t.co/9DGpEX0js", "display_url": "t.co/9DGpEX0js"}], "hashtags": [{"text": "#DolceAmoreHotSeat"}], "symbols": [{"text": "#"}]}, "favorited": false, "retweeted": false, "possibly_sensitive": false, "filter_level": "low", "lang": "en"}]}]
```

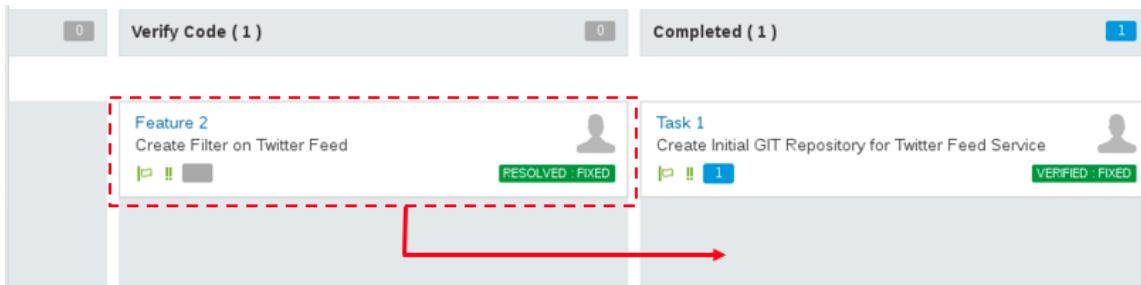
- Now change the appended URL to **/statictweets/alpha** and **press enter**. This will cause records containing the text **alpha** in the tweet's text or hashtags to be returned.

```
{
  "tweets": [
    {
      "created_at": "Tue Aug 16 19:09:09 +0000 2016",
      "id": 765626469619859456,
      "id_str": "765626469619859456",
      "in_reply_to_status_id": null,
      "in_reply_to_status_id_str": null,
      "in_reply_to_user_id": null,
      "in_reply_to_user_id_str": null,
      "in_reply_to_screen_name": null,
      "lang": "en",
      "long_form_url": "https://twitter.com/nickcourtney/status/765626469619859456",
      "name": "Nick Courtney",
      "profile_background_color": "F0F0F0",
      "profile_background_image": "http://pbs.twimg.com/profile_background_images/23947154/111.jpg",
      "profile_background_image_url": "https://pbs.twimg.com/profile_background_images/23947154/111.jpg",
      "profile_banner": "http://pbs.twimg.com/profile_banners/23947154/111",
      "profile_banner_url": "https://pbs.twimg.com/profile_banners/23947154/111",
      "profile_color": "333333",
      "profile_image": "http://pbs.twimg.com/profile_images/23947154/111_normal.jpg",
      "profile_image_url": "https://pbs.twimg.com/profile_images/23947154/111_normal.jpg",
      "profile_link_color": "000000",
      "profile_sidebar_fill_color": "E0E0E0",
      "profile_text_color": "333333",
      "profile_use_background_image": true,
      "screen_name": "NickCourtney",
      "text": "I've had pretty good service at Alpha Of iPhone",
      "truncated": false
    }
  ]
}
```

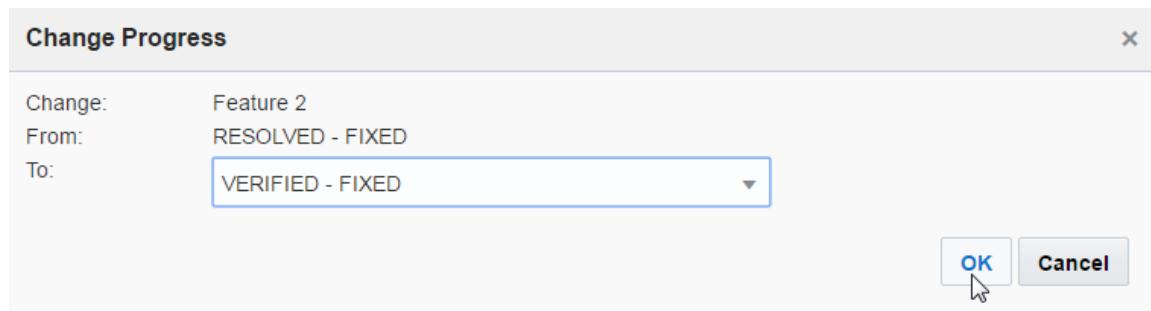
- To complete the Sprint Feature, click **Agile** on left hand navigation. Then click on the **Active Sprints** button.



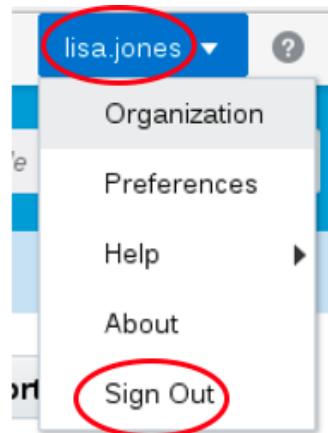
- Complete the feature request by Dragging and Dropping **Feature 2** (Create Filter on Twitter Feed) from the **Verify** Column to the **Completed** Column.



- Set the Status to **VERIFIED – FIXED** and click **OK**



- Sign Out** as Lisa Jones



- You are now done with this lab.

Supplementary Assignment – Twitter Live Feed Credentials

Create Twitter App

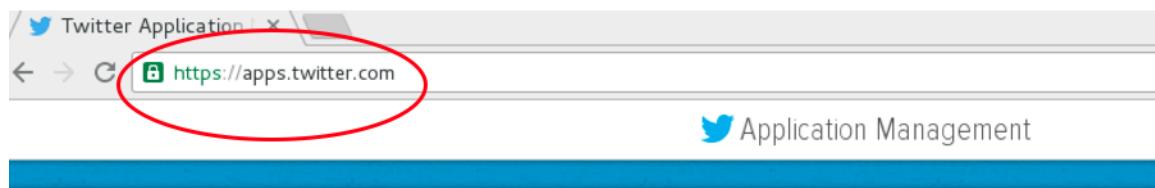
This is an optional assignment, during which you'll have an opportunity to put your new knowledge of the Developer Cloud Service to work by extending our static twitter microservices to use live twitter data. In this exercise, you will acquire Twitter Application Credentials and use them to operate on a live twitter feed in your microservices. For the purposes of this assignment, you will use a personal account to log in to twitter and generate the credentials. However, in the context of our application, assume that these credentials have been provided by Product Management and represent the approved credentials for our production application.

You have two options for managing this code change in the version control system. If you would like more practice with the multi-user workflow, you can start a new branch for this feature, commit to that branch, create a merge request, and approve the merge. We'll refer to this in the instructions as **Method A**. If you're comfortable with that workflow, you can switch to master in your local repository, pull the latest revision from the cloud, and commit and push directly to master for this exercise. This will be **Method B**.

STEP 25: Create New Twitter App

To generate the unique twitter credentials for our microservices, we need to sign in to twitter and create a new application for this project, then generate access tokens for it.

- Navigate to <https://apps.twitter.com>. Click on the **Sign In** link.



Twitter Apps

Please [sign in](#) with your Twitter Account to create :

- If you are already a twitter user, **Log In** using your twitter credentials.
Otherwise, click on the **Sign up Now** link

Log in to Twitter

Phone, email or username

Password

Log in

Remember me · [Forgot password?](#)

New to Twitter? [Sign up now »](#)

Already using Twitter via text message? [Activate your account »](#)

- Once logged in, click on the **Create New App** button.

 Application Management

Twitter Apps

You don't currently have any Twitter Apps.

[Create New App](#)

- Enter the following** and Click on the **Create your Twitter application button**. When entering the Application Name, append something unique to the Name's end. E.g. your initials or name.

Name : **JavaTwitterMicroservice<UniqueName>**
Description : **A Twitter Feed Microservice**
Website : **https://cloud.oracle.com/acc**
Developer Agreement: **Click Yes**

Create an application

Application Details

Name *

Your application name. This is used to attribute the source of a tweet and in user-facing authorization screens. 32 characters max.

Description *

Your application description, which will be shown in user-facing authorization screens. Between 10 and 200 characters max.

Website *

Your application's publicly accessible home page, where users can go to download, make use of, or find out more information about your application. This fully-qualified URL must be a valid URL. If you don't have a URL yet, just put a placeholder here but remember to change it later.

Callback URL

Where should we return after successfully authenticating? OAuth 1.0a applications should explicitly specify their oauth_callback URL on the request token step, regardless of whether they plan to use callbacks. If you don't have a URL yet, leave this field blank.

Developer Agreement
 Yes, I have read and agree to the [Twitter Developer Agreement](#).

Create your Twitter application

- Click on the **Keys and Access Tokens** tab.

JavaTwitterMicroservicePCD

Details Settings **Keys and Access Tokens** Permissions

Application Settings

Keep the "Consumer Secret" a secret. This key should never be human-readable in your application.

Consumer Key (API Key) XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

Consumer Secret (API Secret) XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

Access Level Read and write ([modify app permissions](#))

Owner username

Owner ID 1234567

- If at the bottom of the page your Tokens are not visible, click on the **Create my access tokens button**

Your Access Token

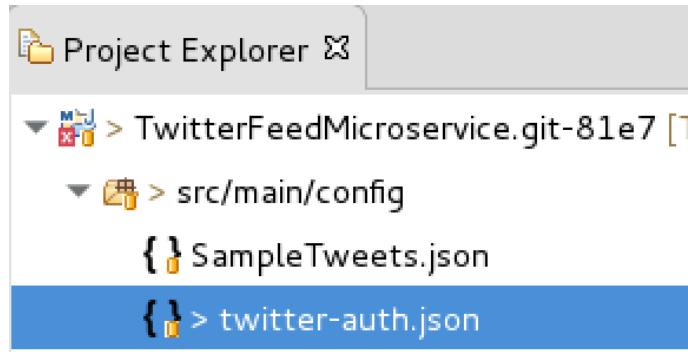
You haven't authorized this application for your own account yet.

By creating your access token here, you will have everything you need to use your application's current permission level.

Token Actions

[Create my access token](#)

- Note: If you are following **Method B**, before you start modifying code in Eclipse, you should switch to the master branch and pull from the remote repository.
- Return to Eclipse, and in the Project Explorer tab, expand **TwitterFeedMicroservices.git > src/main/config** and double click on **twitter-auth.json** to load the source.



- This is the File that will be deployed to the Application Container Cloud. Edit this file by replacing the xxx's in **consumerKey**, **consumerSecret**, **token** and **tokenSecred** with the **Consumer Key (API Key)**, **Consumer Secret (API Secret)**, **Access Token** and **Access Token Secret** found on the Twitter Application Management page.

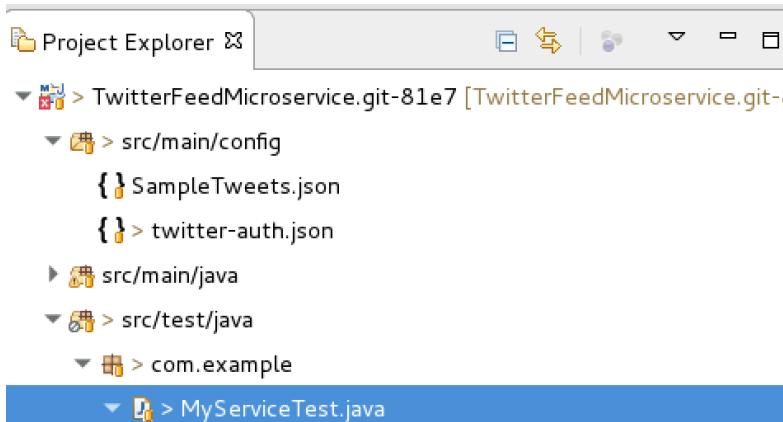
The screenshot shows the Eclipse IDE's code editor with the 'twitter-auth.json' file open. The file contains a single object with four properties: 'consumerKey', 'consumerSecret', 'token', and 'tokenSecret', each assigned a long string of characters. These strings are highlighted in blue, likely indicating they are placeholder values that need to be replaced with actual API keys and tokens.

```
{ } twitter-auth.json <input>
{
    "consumerKey": "xxxxxxxxxxxxxxxxxxxxxxxxxxxx",
    "consumerSecret": "xxxxxxxxxxxxxxxxxxxxxxxxxxxx",
    "token": "xxxxxxxxxxxxxxxxxxxxxxxxxxxx",
    "tokenSecret": "xxxxxxxxxxxxxxxxxxxxxxxxxxxx"
}
```

- Click on the Save All icon in Eclipse 
- So we can test locally, let's repeat the same step by updating the Test Code's twitter-auth.json credentials. Open the file located in **TwitterFeedMicroservices.git > src/test/resources > twitter-auth.json** and update. Once updated, click on the **Save All** Icon.



- Let's now un-comment the code that will allow the online Twitter Feed to be tested. Using the Project Explorer, open the **TwitterFeedMicroservice.git > src/test/java > com.example > MyServiceTest.java** file.



- In the MyServiceTest.java file, located the method **testGetTweets()** and **remove the comments** surrounding that method.

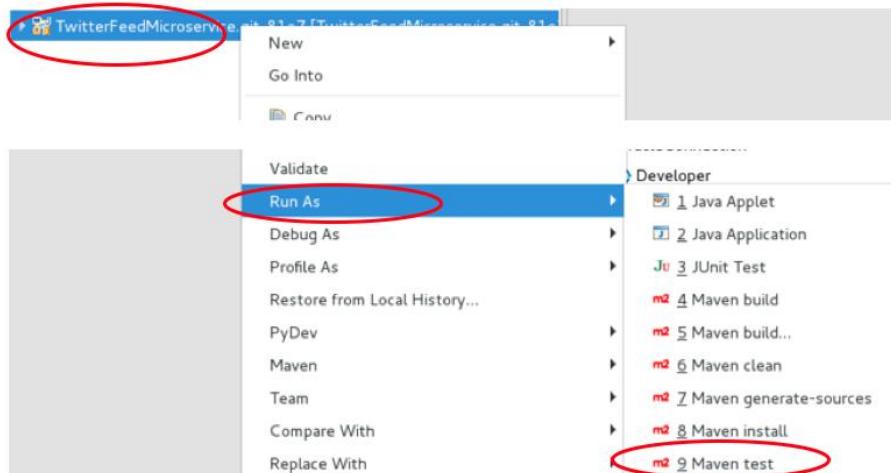
```
    @Test
    public void testGetStaticTweets() {
        String responseMsg = target.path("statictweets").request().get(String.class);
        assertNotNull(responseMsg);
    }

    @Test
    public void testGetStaticSearchTweets() {
        String responseMsg = target.path("statictweets/alpha").request().get(String.class);
        assertNotNull(responseMsg);
    }

    /* @Test
    public void testGetTweets() {
        String responseMsg = target.path("tweets").request().get(String.class);
        assertNotNull(responseMsg);
    }
    */

```

- Click on the Eclipse Save All icon 
- Run the Test by right clicking on **TwitterFeedMicroservice** and selecting **Run As > Maven Test**



- After the tests run, the testGetTweets() method will return the message "The client read 10 messages!," and all Tests should complete successfully.

```
-----  
T E S T S  
-----  
Running com.example.MyServiceTest  
Aug 18, 2016 11:40:40 AM org.glassfish.grizzly.http.server.NetworkLister  
INFO: Started listener bound to [localhost.localdomain:8080]  
Aug 18, 2016 11:40:40 AM org.glassfish.grizzly.http.server.HttpServer  
INFO: [HttpServer] Started.  
{"created_at": "Thu Aug 18 15:40:38 +0000 2016", "id": 76629877038449459},  
{"created_at": "Thu Aug 18 15:40:38 +0000 2016", "id": 76629877040546611},  
{"created_at": "Thu Aug 18 15:40:38 +0000 2016", "id": 76629877039278899},  
{"created_at": "Thu Aug 18 15:40:38 +0000 2016", "id": 766298770376052736},  
{"delete": {"status": {"id": 678115516905574401, "id_str": "678115516905574401"}},  
{"delete": {"status": {"id": 335505648538234881, "id_str": "335505648538234881"}},  
{"created_at": "Thu Aug 18 15:40:38 +0000 2016", "id": 766298770405482496},  
{"created_at": "Thu Aug 18 15:40:38 +0000 2016", "id": 76629877040952115},  
{"created_at": "Thu Aug 18 15:40:38 +0000 2016", "id": 766298770392965120},  
{"created_at": "Thu Aug 18 15:40:38 +0000 2016", "id": 76629877039713075},  
The client read 10 messages!  
Aug 18, 2016 11:40:42 AM org.glassfish.grizzly http server NetworkLister
```

- If you're following **Method A**, now that you've enabled this new feature to access the live twitter feed, you can follow the previous steps used in this document to commit the code to the cloud. Once committed, you will use the Developer Cloud Service to create a merge request and then approve that request. Once the master branch is updated, an automatic build and deployment to the Application Container Cloud Service will be performed. Verify that deployment is successful before continuing.
 - If you're following **Method B**, now that you've enabled this new feature to access the live twitter feed, you can follow the previous steps used in this document to commit the code to the cloud. That will trigger an automatic build and cause the Application Container Cloud Service deployment to be performed by the Developer Cloud Service. Verify that deployment is successful before continuing.
 - For either method, you will now be able append **/tweets** to the end of the Application Container Cloud Service URL and retrieve the Live Tweets.
 - The example below shows the live tweets returned, once the application is re-deployed.

```
← → C https://javatwittermicroservice-gse00002055.apaas.em2.oraclecloud.com/tweets

{"tweets": [{"delete": {"status": {"id": 628210773559357440, "id_str": "628210773559357440", "user_id": 766301530232328192, "id_str": "766301530232328192, "created_at": "Thu Aug 18 15:51:36 +0000 2016", "id": 628210773559357440, "id_str": "628210773559357440, "user_id": 766301530232328192, "created_at": "Thu Aug 18 15:51:36 +0000 2016, "source": ": \"http://twitter.com/\" rel=\"nofollow\", "truncated": false, "in_reply_to_status_id": null, "in_reply_to_status_id": null, "in_reply_to_user_id": null, "in_reply_to_user_id": null, "in_reply_to_screen_name": null, "in_reply_to_screen_name": null, "geo": null, "coordinates": null, "place": null, "contributors": null, "is_quote_status": false, "extended_tweet": null, "extended_entities": null, "entities": {"text": "Client\u003c/a\u003e", "hashtag": null, "url": null, "symbol": null}, "favorited": false, "retweeted": false, "possibly_sensitive": false, "lang": "en"}, "screen_name": "irakliytoles", "location": "Tbilisi, Georgia", "name": "Ira", "url": "http://t.co/1DfXWzJL", "profile_image_url": "http://pbs.twimg.com/profile_images/1720338750/537_normal.gif", "profile_banner_url": "http://pbs.twimg.com/profile_banners/1720338750/1438050000", "profile_text_color": "#333333", "profile_use_background_image": true, "profile_background_color": "#EFEFEF", "profile_link_color": "#000000", "profile_sidebar_fill_color": "#EFEFEF", "profile_sidebar_border_color": "#EFEFEF", "profile_background_image_url": "https://abs.twimg.com/images/themes/theme14/bg.gif"}]}]
```

Appendix 2 – Installing Eclipse

Download and Install Eclipse

In this appendix you will download and install Eclipse containing the Oracle Plugin.

STEP 26: Download Eclipse

- Go to the following URL: <http://www.oracle.com/technetwork/developer-tools/eclipse/downloads/index.html>
- Accept the **licensing agreement**, and then select the **Neon** version of Eclipse required for your operating system.

Oracle Technology Network > Developer Tools > Enterprise Pack for Eclipse > Downloads

Overview Downloads Documentation Community Learn More

Oracle Enterprise Pack for Eclipse (12.2.1.3.1)

You must accept the OTN License Agreement for Oracle Enterprise Pack for Eclipse to download this software.

Accept License Agreement Decline License Agreement

Network Installer

Create a custom installation of Eclipse with Oracle Tools, by choosing the desired OEPE release and the required capabilities.

[Launch Network Installer](#)

Packaged Distributions

These distributions include Eclipse with Oracle Tools already installed. Just download and unzip.

	Mars	Neon
Windows 64-bit	Download	Download
Linux 64-bit	Download	Download
Mac OS X 64-bit	Download	Download

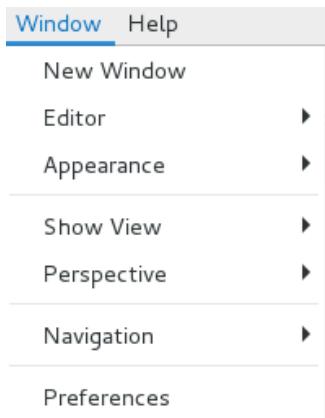
- Once you've downloaded eclipse, extract the zip file and install.

Optionally Configure Proxies

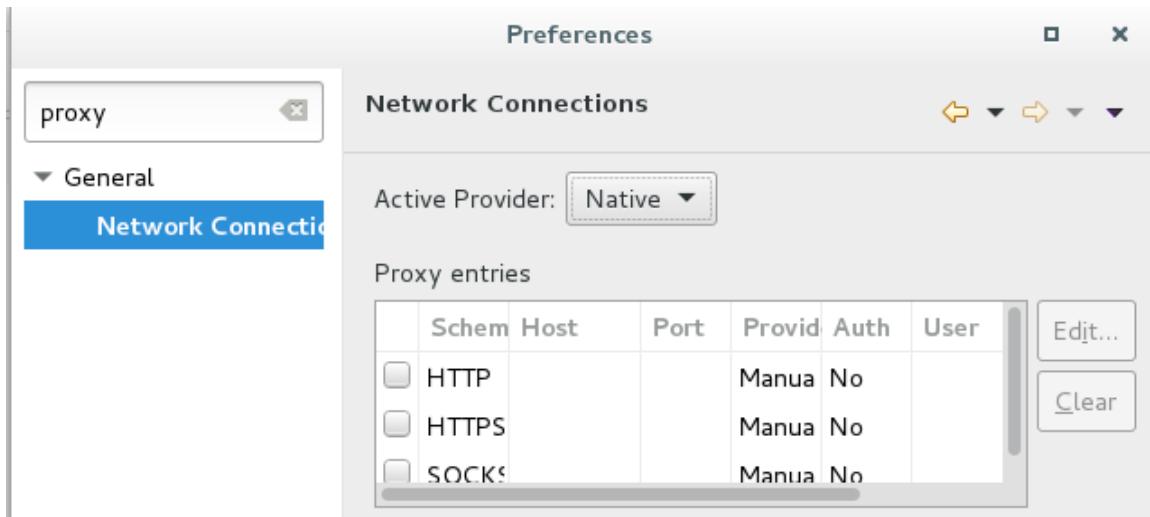
STEP 27: Configuring Proxies

If you are running Eclipse behind a firewall and need to configure the proxy setting, there are several updates to be made. First, you need to ensure that Eclipse's proxy is set, next you need to update the maven proxy setting, then finally, you need to ensure that the Oracle Plugin will work with your proxy settings.

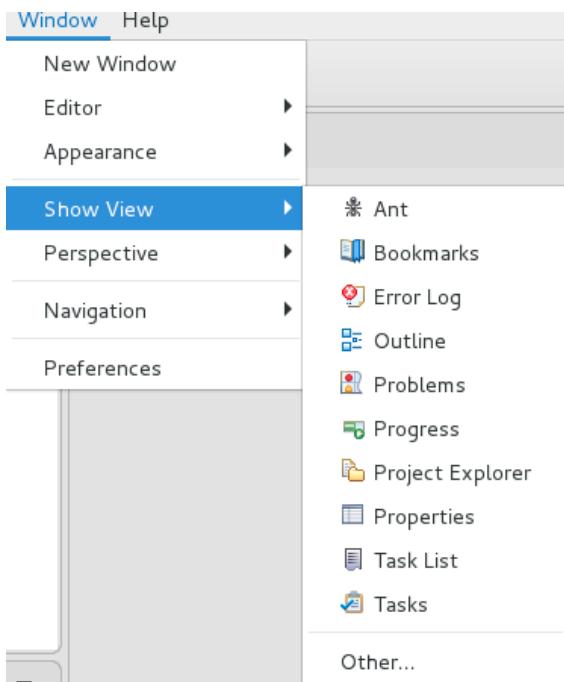
- To set configure Eclipse's proxy, open Eclipse and navigate to the Properties menu. Depending on the operating system, this drop down is found either from the **Eclipse > Preferences**, or **Window > Preferences**



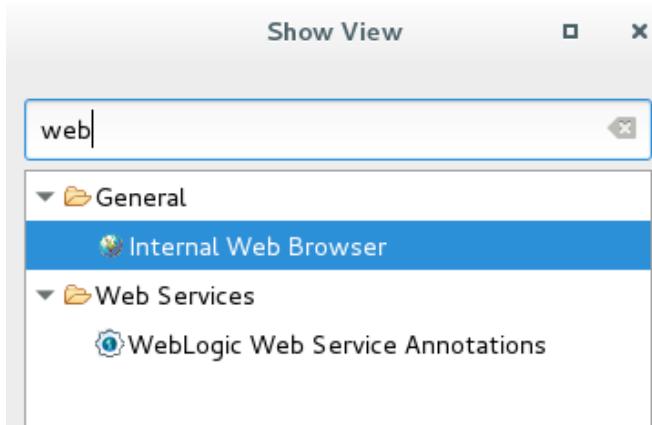
- From the preferences panel, enter “**proxy**” into the search window, and click on **Network Connections**. Select **Native** for the Active Provider. This setting works well, but it requires that you have the proxy setting configured correctly on the system running Eclipse – e.g. Windows, MAC OS or Linux. Selecting Manual should also work, but some of the plugins require the underlying operating system’s proxy to be configured.



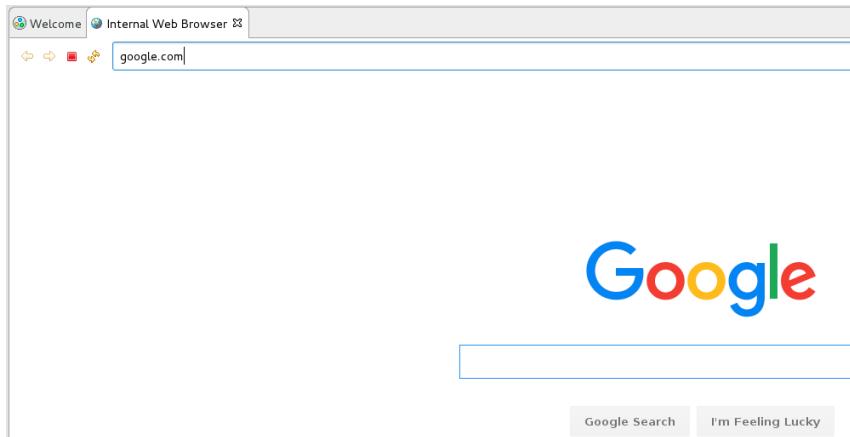
- To test that your connection works, select the menu option **Window > Show View > Other**



- Type “**web**” in the search field, **select Internal Web Browser** and click on **OK**

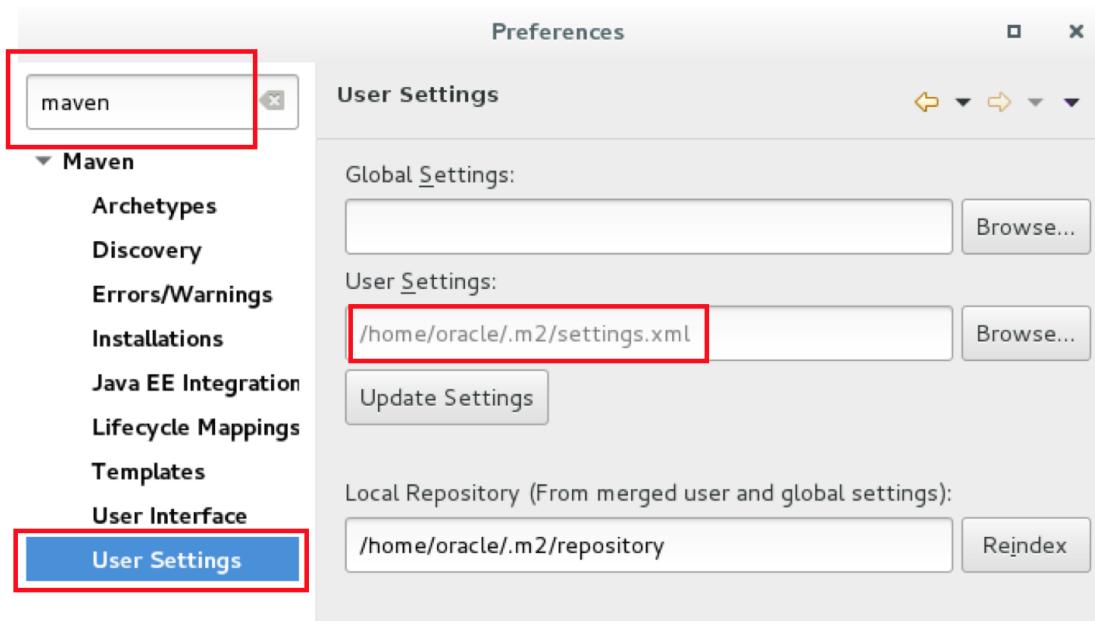


- Enter a **URL** into the and **press enter** to test your proxy settings.



STEP 28: Update the Eclipse / Maven proxy

- From the **Eclipse > Preference** or **Window > Preferences** panel, enter **Maven** into the search box. Click on the Maven **User Settings**. Make note of the directory where the settings.xml file is to be located. In the example below, the Maven User Settings are to be located in the **/home/oracle/.m2** directory



- Close Eclipse
- If the directory does not exist where the settings.xml file is to be located, **create the directory**. In this example, we will create the .m2 directory. Also, create the settings.xml file, if it does not exist. Add the following to the settings.xml file (NOTE: you will need to use your correct **Host, Port, nonProxyHosts, username and Password** settings):

```
<?xml version="1.0" encoding="UTF-8"?>
<settings xmlns="http://maven.apache.org/SETTINGS/1.1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.1.0
  http://maven.apache.org/xsd/settings-1.1.0.xsd">
  <proxies>
    <proxy>
      <active>true</active>
      <protocol>http</protocol>
      <username>proxyuser</username>
      <password>proxypass</password>
      <host>www-proxy.us.oracle.com</host>
      <port>80</port>
      <nonProxyHosts>local.net|some.host.com</nonProxyHosts>
    </proxy>
    <proxy>
      <active>true</active>
      <protocol>https</protocol>
      <username>proxyuser</username>
      <password>proxypass</password>
      <host>www-proxy.us.oracle.com</host>
      <port>80</port>
      <nonProxyHosts>local.net|some.host.com</nonProxyHosts>
    </proxy>
  </proxies>
</settings>
```

- Reload Eclipse to use the new maven settings