

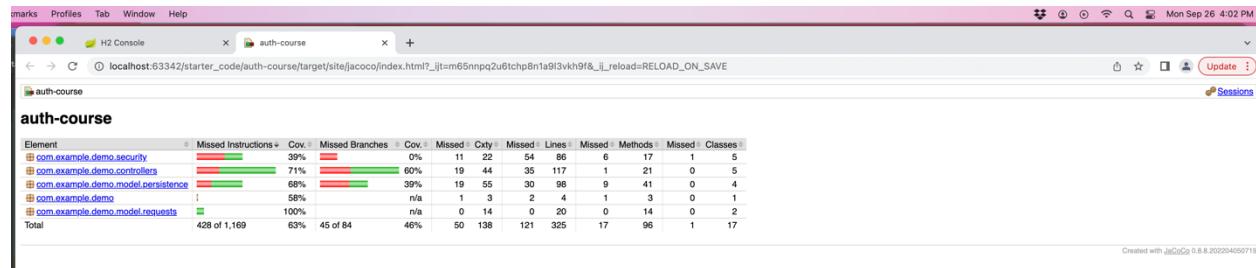
## Screenshots for eCommerce Application Submission

Build the project, tests, and test coverage:.....	2
Postman requests.....	3
Logging and Splunk set up.....	4
Splunk query example .....	5
Splunk dashboard example .....	6
Splunk alerts example .....	7
AWS Set-up.....	10
Jenkins build .....	11
Tomcat automatic deployment of Jenkins build artifact.....	13

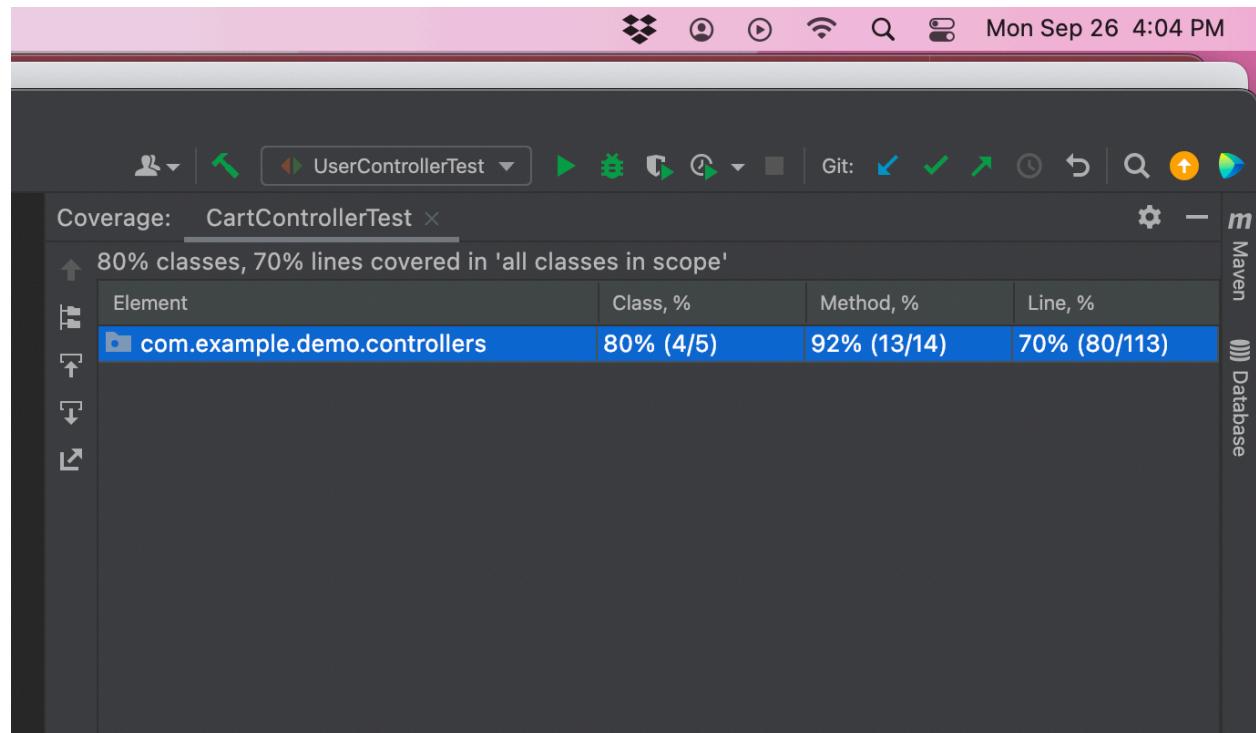
## Build the project, tests, and test coverage:

Using “mvn clean package”, all the test classes will be run, and a coverage report will be created by the Jacoco plugin.

After the build, this coverage report is found at target/site/jacoco/index.html, and shows a total project coverage of 63%.



In particular, the src package com.example.demo.controllers (where most of my code was written) has a class coverage of 80%, a method coverage of 92%, and a line coverage of 70%.



## Postman requests

Once the API is running, you can use the Postman request collection, found at `src/main/resources/ecommerce.postman_collection.json`, to test the API endpoints with security. Once a user is created and logged in, their authorization header (the encoded JWT) must be present for requests to the user, cart, and order endpoints. The item endpoints do not require authentication.

The screenshot shows the Postman application interface. On the left, the sidebar displays 'My Workspace' with a collection named 'ecommerce'. Under 'ecommerce', several requests are listed: POST CreateUserSusan, POST CreateUserSteve, POST LoginSusan, POST LoginSteve, GET GetUserSusan, GET GetUserSteve, GET GetUserId1, POST AddToCartSusan, POST AddToCartSusanMore, POST RemoveFromCartSusan, POST SubmitAnOrderSusan, GET GetOrderHistorySusan, GET GetItems, GET GetItem1, and GET GetItemRoundWidget. The 'CreateUserSusan' request is selected and expanded. The main workspace shows a POST request to `http://localhost:8082/api/user/create`. The 'Body' tab is selected, showing a JSON payload:

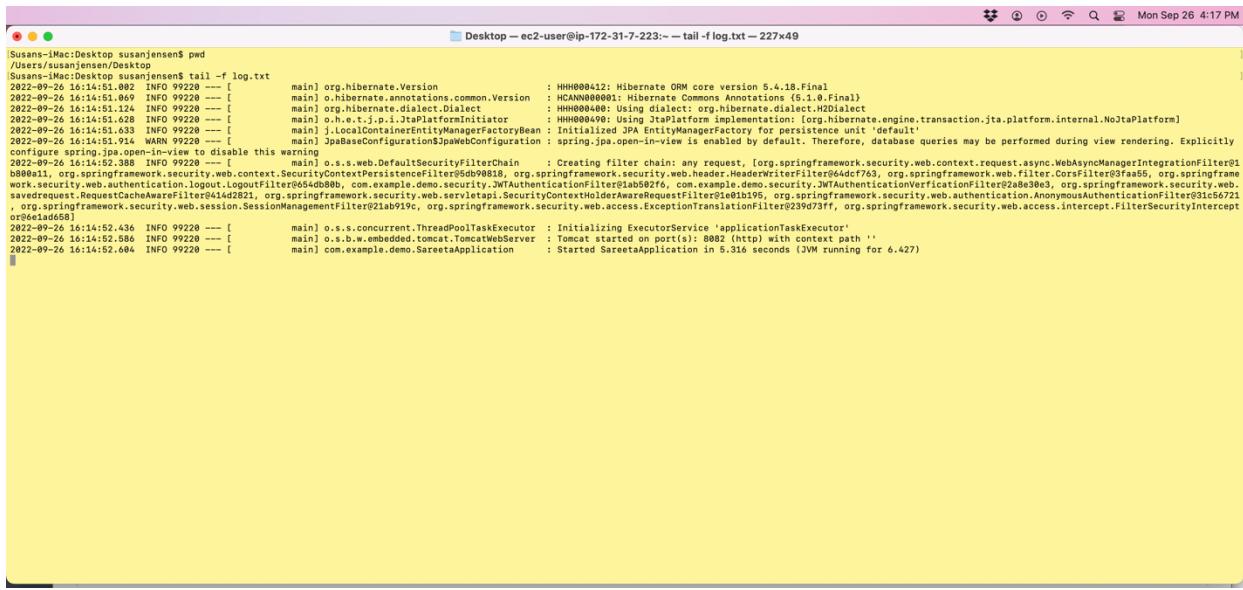
```
1
2   ...
3     "username": "Susan",
4     ...
5     "password": "abcdefg",
6     ...
7     "confirmPassword": "abcdefg"
```

Below the body, the 'Test Results' section shows a successful response with status 200 OK, time 694 ms, and size 439 B. The response body is:

```
1
2   ...
3     "id": 1,
4     ...
5     "username": "Susan"
```

## Logging and Splunk set up

With the API running locally in my IDE, I specified the logs to be written locally to “/Users/susanjensen/Desktop/log.txt”. Here is the tail of the log file, after starting the app:



```
Susan's-Mac:Desktop susanjensen$ pwd
/Users/susanjensen/Desktop
Susan's-Mac:Desktop susanjensen$ tail -f log.txt
2022-09-26 16:14:51.889 INFO 99220 --- [main] org.hibernate.Version : HHH000412: Hibernate ORM core version 5.4.18.Final
2022-09-26 16:14:51.889 INFO 99220 --- [main] o.hibernate.annotations.common.Version : HCANN000001: Hibernate Commons Annotations (5.1.0.Final)
2022-09-26 16:14:51.124 INFO 99220 --- [main] org.hibernate.dialect.Dialect : HHH000400: Using dialect: org.hibernate.dialect.H2Dialect
2022-09-26 16:14:51.628 INFO 99220 --- [main] o.h.e.t.j.p.i.JtaPlatformInitiator : HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
2022-09-26 16:14:51.628 INFO 99220 --- [main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
2022-09-26 16:14:51.714 WARNING 99220 --- [main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries may be performed during view rendering. Explicitly configure spring.jpa.open-in-view to disable this warning
2022-09-26 16:14:52.388 INFO 99220 --- [main] o.s.web.DefaultSecurityFilterChain : Creating filter chain: any request, [org.springframework.security.web.context.request.async.WebAsyncManagerIntegrationFilter@1b8800, org.springframework.security.web.header.HeaderWriterFilter@1c05d4, org.springframework.security.web.context.HttpSessionEventPublisherFilter@11a333, org.springframework.security.web.context.request.asyncWebRequestFilter@8414d2821, org.springframework.security.web.authentication.logout.LogoutFilter@8414d2809, com.example.demo.security.JwtAuthenticationFilter@ab9874, com.example.demo.security.JwtAuthenticationFilter@2adea30e3, org.springframework.security.web.savedrequest.RequestCacheAwareFilter@414d2821, org.springframework.security.web.servletapi.SecurityContextHolderAwareRequestFilter@e1eb1b95, org.springframework.security.web.authentication.UsernamePasswordAuthenticationFilter@831864721, org.springframework.security.web.session.SessionManagementFilter@21a1919c, org.springframework.security.web.access.ExceptionTranslationFilter@239d73ff, org.springframework.security.web.access.ExceptionTranslationFilter@239d73ff, org.springframework.security.web.access.intercept.FilterSecurityInterceptor@0e61ad68]
2022-09-26 16:14:52.436 INFO 99220 --- [main] o.s.s.concurrent.ThreadPoolTaskExecutor : Initializing ExecutorService 'applicationTaskExecutor'
2022-09-26 16:14:52.586 INFO 99220 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8082 (http) with context path ''
2022-09-26 16:14:52.694 INFO 99220 --- [main] com.example.demo.SareetaApplication : Started SareetaApplication in 5.316 seconds (JVM running for 6.427)
```

In Splunk, running locally, I added a data source, which is continuous monitoring of this log file.

## Splunk query example

I want to see all the messages so far logged, just from the UserController class:

```
index=_* OR index=* sourcetype=ecommerce_log "c.e.demo.controllers.UserController"
```

The screenshot shows a Splunk search interface with the following details:

- Search Bar:** index=\_\* OR index=\* sourcetype=ecommerce\_log "c.e.demo.controllersUserController"
- Results:** 2 events (9/25/22 4:00:00.000 PM to 9/26/22 4:58:30.000 PM) No Event Sampling
- Event List:**

Time	Event
9/26/22 4:24:14.073 PM	INFO 99220 --- [nio-8882-exec-3] c.e.demo.controllers UserController : User was created: name set with Steve
9/26/22 4:24:08.062 PM	INFO 99220 --- [nio-8882-exec-1] c.e.demo.controllers UserController : User was created: name set with Susan
- Selected Fields:** host 1, source 1, sourcetype 1
- Interesting Fields:** date\_hour 1, date\_minute 1, date\_month 1, date\_second 1, date\_wday 1, date\_year 1, date\_zone 1, index 1, linecount 1

A little while later, the same search returns more events, so I know the continuous monitoring is working correctly.

The screenshot shows a Splunk search interface with the following details:

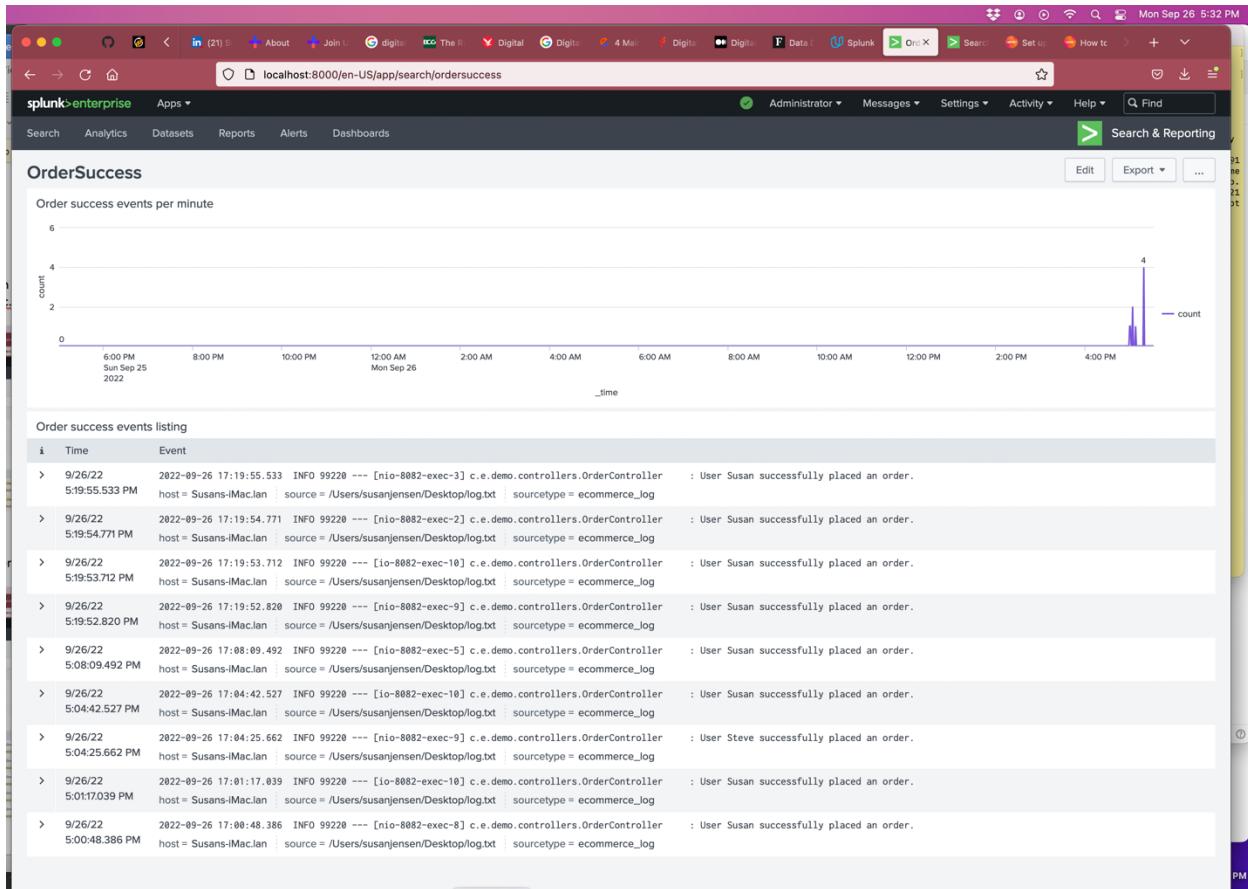
- Search Bar:** index=\_\* OR index=\* sourcetype=ecommerce\_log "successfully placed an order"
- Results:** 5 events (9/25/22 5:00:00.000 PM to 9/26/22 5:10:29.000 PM) No Event Sampling
- Event List:**

Time	Event
9/26/22 5:08:09.492 PM	INFO 99220 --- [nio-8882-exec-5] c.e.demo.controllers.OrderController : User Susan successfully placed an order.
9/26/22 5:04:42.527 PM	INFO 99220 --- [nio-8882-exec-10] c.e.demo.controllers.OrderController : User Susan successfully placed an order.
9/26/22 5:04:25.662 PM	INFO 99220 --- [nio-8882-exec-9] c.e.demo.controllers.OrderController : User Steve successfully placed an order.
9/26/22 5:01:17.039 PM	INFO 99220 --- [nio-8882-exec-10] c.e.demo.controllers.OrderController : User Susan successfully placed an order.
9/26/22 5:00:48.386 PM	INFO 99220 --- [nio-8882-exec-8] c.e.demo.controllers.OrderController : User Susan successfully placed an order.
- Selected Fields:** host 1, source 1, sourcetype 1
- Interesting Fields:** date\_hour 1, date\_minute 1, date\_month 1, date\_second 1, date\_wday 1, date\_year 1, date\_zone 1, index 1, linecount 1

## Splunk dashboard example

I'd like a dashboard that lists the successful orders from my log, as well as charting the number of successful orders over time, counting successful order per minute. I will add "min" and "max" values to my chart.

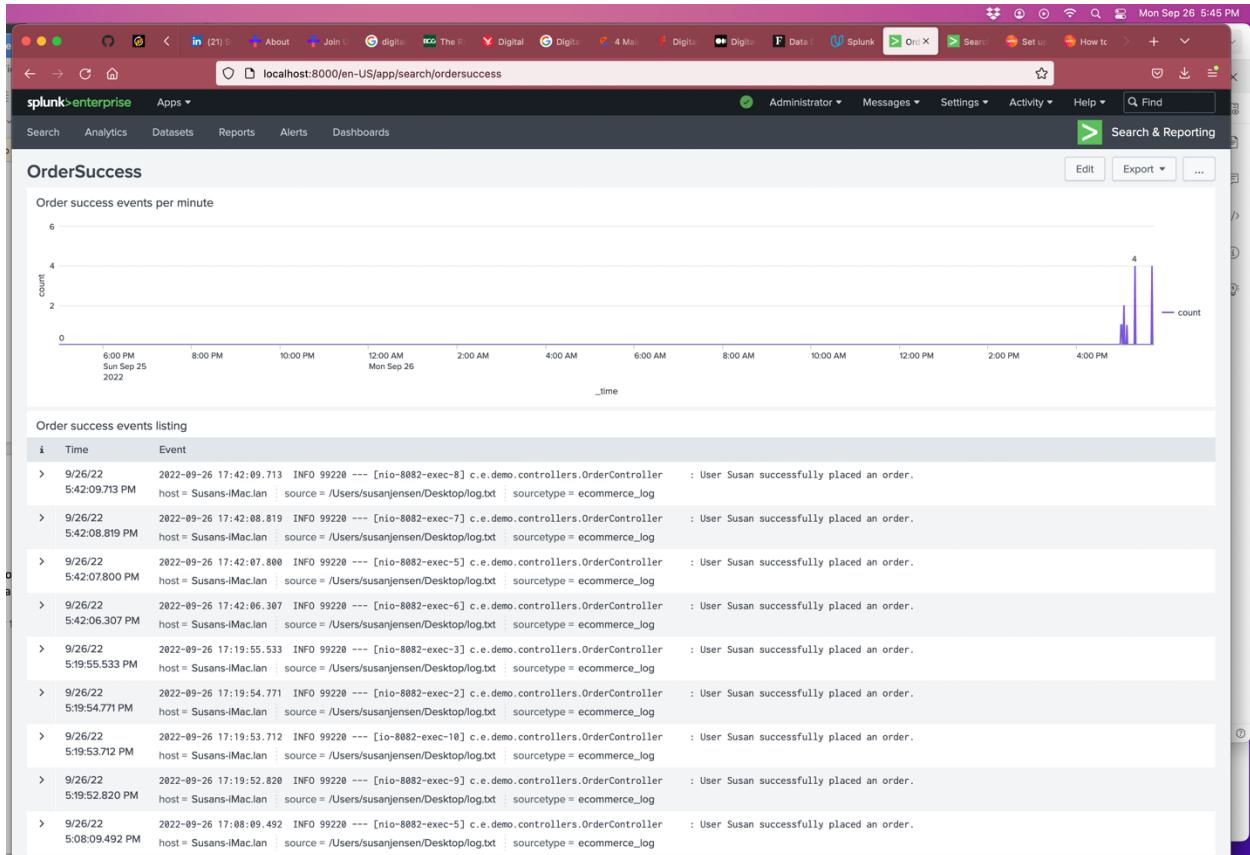
Below we can see that the max number of successful orders (in the chart) was four, and we can see from the order listing that user Susan placed four orders in the minute of 5:19 pm.



## Splunk alerts example

Now that I have a dashboard showing the successful orders, I want an alert to trigger automatically if I don't have any successful orders in a ten-minute period.

I had four successful orders at 5:19 pm, then another four successful orders at 5:42 pm (see screenshot, below)



I set up my alert at about 5:38 pm, and it started to trigger once per minute, starting at 5:39 pm (see screenshot, below).

**Order success drops to zero**

Enabled: Yes. Disable  
App: search  
Permissions: Private. Owned by sujensen. Edit  
Modified: Sep 26, 2022 5:41:44 PM  
Alert Type: Real-time. Edit

Trigger Condition: Number of Results is = 0 in 10 minutes. Edit  
Actions: 1 Action Edit  
Add to Triggered Alerts

**Trigger History**  
20 per page ▾

TriggerTime	Actions
1 2022-09-26 17:41:32 PDT	<a href="#">View Results</a>
2 2022-09-26 17:40:27 PDT	<a href="#">View Results</a>
3 2022-09-26 17:39:21 PDT	<a href="#">View Results</a>

Since I had some successful orders at 5:42 pm, I don't expect the alert to be triggered again until about 5:52 pm. And, that is exactly what happened, as I expected. The alert was triggered when there hadn't been any successful orders for 10 minutes:

**Order success drops to zero**

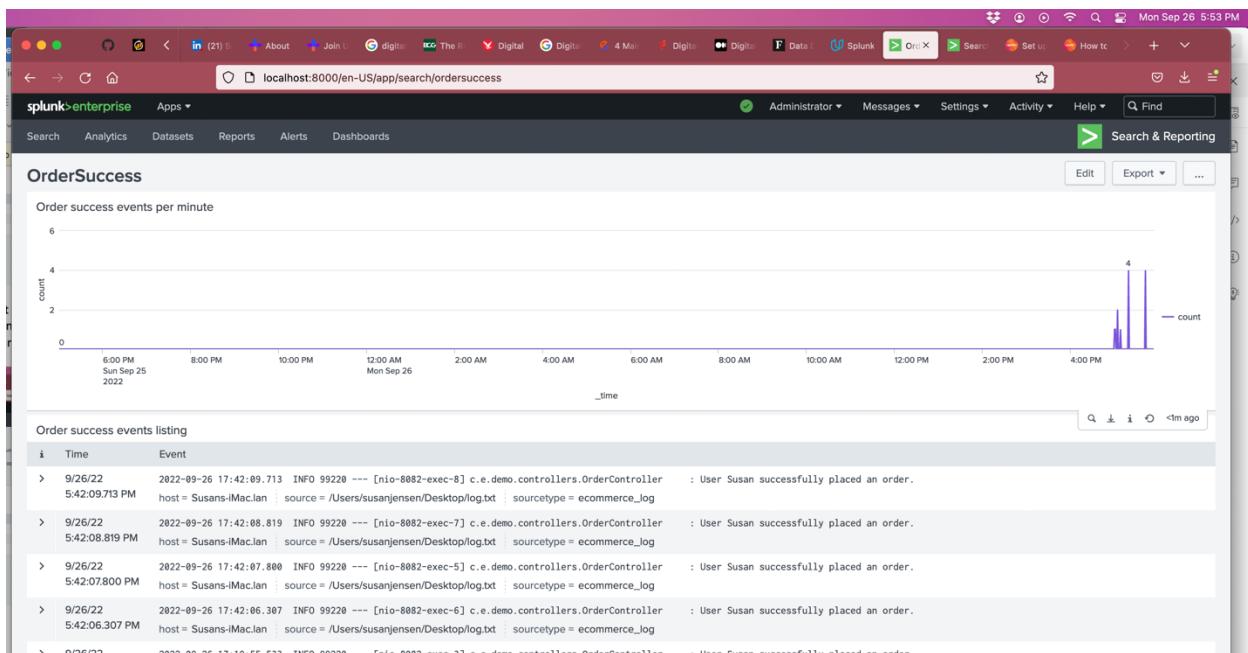
Enabled: Yes. Disable  
App: search  
Permissions: Private. Owned by sujensen. Edit  
Modified: Sep 26, 2022 5:41:44 PM  
Alert Type: Real-time. Edit

Trigger Condition: Number of Results is = 0 in 10 minutes. Edit  
Actions: 1 Action Edit  
Add to Triggered Alerts

**Trigger History**  
20 per page ▾

TriggerTime	Actions
1 2022-09-26 17:52:10 PDT	<a href="#">View Results</a>
2 2022-09-26 17:41:32 PDT	<a href="#">View Results</a>
3 2022-09-26 17:40:27 PDT	<a href="#">View Results</a>
4 2022-09-26 17:39:21 PDT	<a href="#">View Results</a>

And the dashboard chart confirms there have still been no successful orders since 5:42 pm:



## AWS Set-up

For this project, I will use an EC2 instance on AWS to run both Jenkins and my app, both in docker containers. I used a t2.medium shape for the compute size, as the free tier crashed multiple times.

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with various navigation options like EC2 Dashboard, EC2 Global View, Events, Tags, Limits, Instances, Images, Elastic Block Store, Network & Security, and Load Balancing. The main area displays a table of instances with one entry:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
firstinstance	i-08da2f6cf8123f99a	Running	t2.medium	2/2 checks passed	No alarms	us-west-2c	ec2-34-216-178-95.us...

Below the table, there's a detailed view for the instance i-08da2f6cf8123f99a. It shows the following details:

Instance: i-08da2f6cf8123f99a (firstinstance)			
Details   Security   Networking   Storage   Status checks   Monitoring   Tags			
<b>Instance summary</b>			
Instance ID i-08da2f6cf8123f99a (firstinstance)	Public IPv4 address 34.216.178.95   open address	Private IPv4 addresses 172.31.7.223	
IPv6 address -	Instance state Running	Public IPv4 DNS ec2-34-216-178-95.us-west-2.compute.amazonaws.com   open address	
Hostname type IP name: ip-172-31-7-223.us-west-2.compute.internal	Private IP DNS name (IPv4 only) ip-172-31-7-223.us-west-2.compute.internal	Elastic IP addresses -	
Answer private resource DNS name IPv4 (A)	Instance type t2.medium	AWS Compute Optimizer finding Opt-in to AWS Compute Optimizer for recommendations.   Learn more	
Auto-assigned IP address 34.216.178.95 [Public IP]	VPC ID vpc-0d1b3ac0915d11677	Auto Scaling Group name -	
IAM Role -	Subnet ID subnet-031fbfdb9041f98c3	Monitoring disabled	
<b>Instance details</b>			
Platform Amazon Linux (Inferred)	AMI ID ami-0c2ab3b8efb09f272	AMI name amzn2-ami-kernel-5.10-hvm-2.0.20220805.0-x86_64-gp2	Termination protection Disabled

The screenshot shows a terminal window titled "2020-11-28\_Java\_Web\_Dev" connected to the EC2 instance at ip-172-31-7-223. The user has run the command "ssh -i firstKeyPair.pem ec2-user@ec2-34-216-178-95.us-west-2.compute.amazonaws.com" and is now in the root directory. The terminal output shows:

```
~/Desktop -- ec2-user@ip-172-31-7-223:~ -- ssh -i firstKeyPair.pem ec2-user@ec2-34-216-178-95.us-west-2.compute.amazonaws.com -- 186x49
Susans-iMac:2020-11-28_Java_Web_Dev susanjensens$ ssh -i "firstKeyPair.pem" ec2-user@ec2-34-216-178-95.us-west-2.compute.amazonaws.com
Last login: Tue Sep 20 21:57:11 2022 from 76.144.20.177
--| ( --| )
--| ( --| / Amazon Linux 2 AMI
--| ( --| ---|---|
https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-7-223 ~]$ [ec2-user@ip-172-31-7-223 ~]$ [ec2-user@ip-172-31-7-223 ~]$ docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
158f955a62c6 tomcat:9-jdk8 "catalina.sh run" 6 days ago Up 6 days 0.0.0.0:8888->8080/tcp, ::8888->8080/tcp myTomcatServer
6b7c4602e8f9 jenkinsci/blueocean "/sbin/tini -- /usr/bin/docker-entrypoint.sh" 6 days ago Up 6 days 0.0.0.0:8080->8080/tcp, ::8080->8080/tcp, 50000/tcp myContainer
[ec2-user@ip-172-31-7-223 ~]$ [ec2-user@ip-172-31-7-223 ~]$
```

## Jenkins build

My Jenkins is running in a docker container in AWS. Here is my Jenkins home page. My project will build and deploy with “myFirstJob”:

The screenshot shows the Jenkins home page with the URL 34.216.178.95:8080. The main content area displays a table of builds. One build, "myFirstJob", is listed with a green checkmark icon, indicating it has last succeeded 6 days 3 hours ago. It has failed twice, with the most recent failure occurring 6 days 3 hours ago. The duration of the last failure was 20 seconds. There are links for "Atom feed for all", "Atom feed for failures", and "Atom feed for just latest builds". On the left sidebar, there are links for "New Item", "People", "Build History", "Project Relationship", "Check File Fingerprint", "Manage Jenkins", "My Views", "Open Blue Ocean", and "New View". Below these are sections for "Build Queue" (No builds in the queue) and "Build Executor Status" (1 Idle, 2 Idle).

I pushed my code to Github at 7:05 pm:

The screenshot shows the GitHub repository page for "sujensen/ecommerce" at https://github.com/sujensen/ecommerce. The repository is public and forked from udacity/nd035-c4-Security-and-DevOps. The "Code" tab is selected, showing the main branch. A message indicates "This branch is 14 commits ahead, 13 commits behind udacity:master." The commit history lists several commits, including "starter\_code" (more tests, and Postman json requests), ".gitignore" (add target to gitignore), "LICENSE.md" (Add License), and "README.md" (Initial commit). On the right side, there is an "About" section with details about the repository being related to the Java Web Developer, Course - Security and DevOps. It also shows statistics like 0 stars, 0 watching, and 315 forks. The "Releases" section indicates no releases have been published.

Which automatically triggered build #4 in Jenkins (since Jenkins is polling my Github repo once per minute):

The screenshot shows a web browser window with the Jenkins interface. The URL in the address bar is `34.216.178.95:8080/job/myFirstJob/`. The page title is "Project myFirstJob".

**Left Sidebar:**

- [Back to Dashboard](#)
- [Status](#) (highlighted)
- [Changes](#)
- [Workspace](#)
- [Build Now](#)
- [Configure](#)
- [Delete Project](#)
- [Favorite](#)
- [Git Polling Log](#)
- [GitHub](#)
- [Open Blue Ocean](#)
- [Rename](#)

**Recent Changes:**

- Last build (#4), 35 sec ago
- Last stable build (#4), 35 sec ago
- Last successful build (#4), 35 sec ago
- Last failed build (#2), 6 days 3 hr ago
- Last unsuccessful build (#2), 6 days 3 hr ago
- Last completed build (#4), 35 sec ago

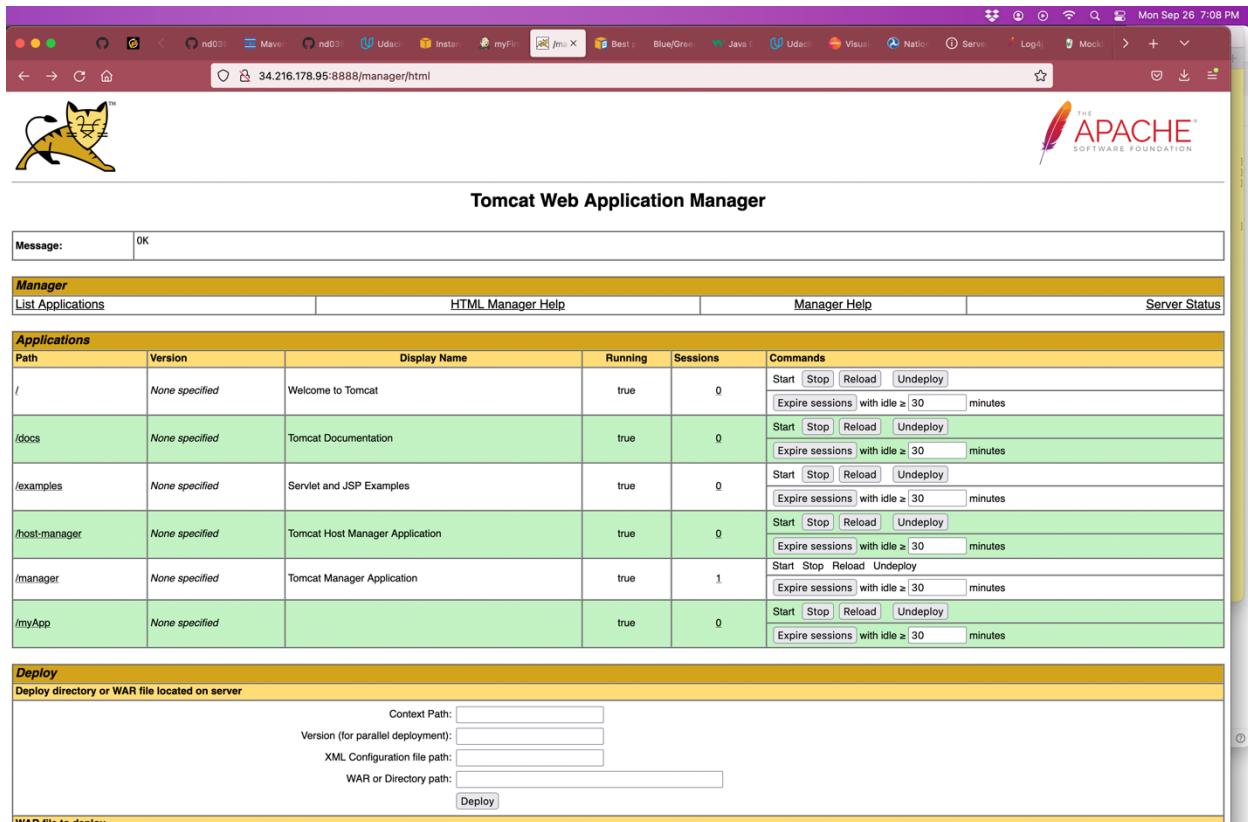
**Build History:**

#	Date
#4	Sep 27, 2022, 2:06 AM
#3	Sep 20, 2022, 10:21 PM
#2	Sep 20, 2022, 10:14 PM
#1	Sep 20, 2022, 10:03 PM

[Atom feed for all](#) [Atom feed for failures](#)

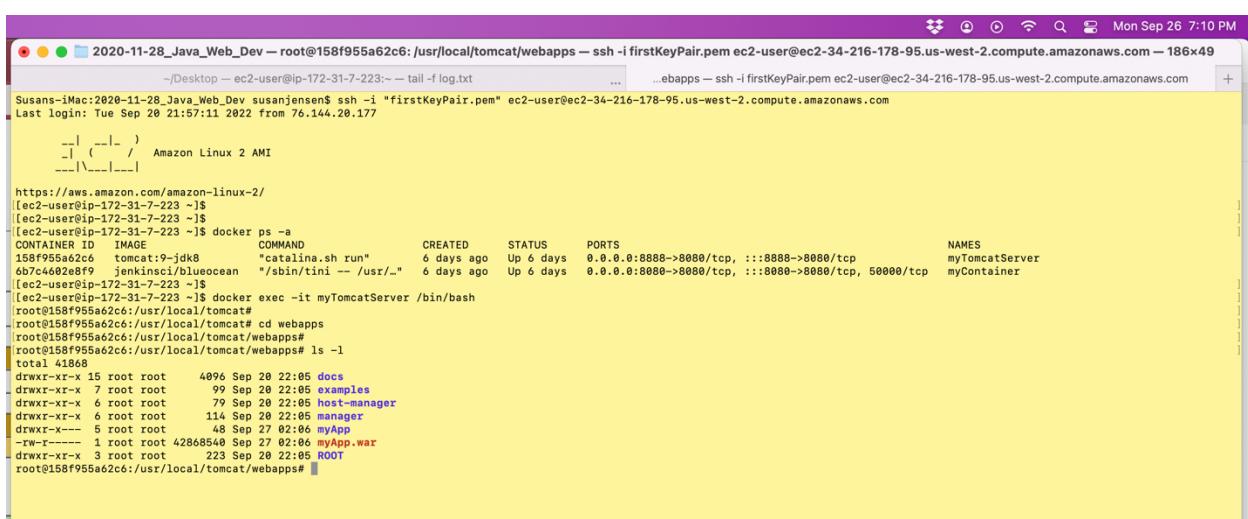
## Tomcat automatic deployment of Jenkins build artifact

In tomcat, accessible from port 8888 on my AWS compute instance, the app is called “myApp”, as seen from the management console:



The screenshot shows the Apache Tomcat Web Application Manager interface. At the top, there's a navigation bar with tabs like 'Manager', 'List Applications', 'HTML Manager Help', 'Manager Help', and 'Server Status'. Below this is a large table titled 'Applications' with columns for 'Path', 'Version', 'Display Name', 'Running', 'Sessions', and 'Commands'. The table lists several applications, including '/docs', '/examples', '/host-manager', '/manager', and '/myApp'. The '/myApp' row shows 'true' under 'Running' and '0' under 'Sessions'. The 'Commands' column for '/myApp' includes 'Start', 'Stop', 'Reload', 'Undeploy', and 'Expire sessions with idle ≥ 30 minutes'. Below the table is a 'Deploy' section with fields for 'Context Path', 'Version (for parallel deployment)', 'XML Configuration file path', 'WAR or Directory path', and a 'Deploy' button. At the bottom, there's a 'WAR file to deploy' section.

Within the tomcat docker container, we can see that “myApp” was updated at the exact same time that the Jenkins build (and post-build promotion) was completed:



```
2020-11-28_Java_Web_Dev root@158f955a62c6:/usr/local/tomcat/webapps -- ssh -i firstKeyPair.pem ec2-user@ec2-34-216-178-95.us-west-2.compute.amazonaws.com -- 186x49
~Desktop - ec2-user@ip-172-31-7-223:~ - tail -f log.txt ... ebapps - ssh -i firstKeyPair.pem ec2-user@ec2-34-216-178-95.us-west-2.compute.amazonaws.com
Last login: Tue Sep 28 21:57:11 2022 from 76.144.20.177
[ec2-user@ip-172-31-7-223 ~]$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS          NAMES
158f955a62c6        tomcat:9-jdk8   "catalina.sh run"   6 days ago         Up 6 days          0.0.0.0:8888->8080/tcp, :::8888->8080/tcp   myTomcatServer
6b7c4402aef9        jenkinsci/blueocean   "/bin/tini -- /usr/..."   6 days ago         Up 6 days          0.0.0.0:8080->8080/tcp, :::8080->8080/tcp, 50000/tcp   myContainer
[ec2-user@ip-172-31-7-223 ~]$ docker exec -it myTomcatServer /bin/bash
root@158f955a62c6:/# cd /usr/local/tomcat/webapps
root@158f955a62c6:/usr/local/tomcat/webapps# ls -l
total 41868
drwxr-xr-x 15 root root    4096 Sep 20 22:05 docs
drwxr-xr-x  7 root root     99 Sep 20 22:05 examples
drwxr-xr-x  6 root root    79 Sep 20 22:05 host-manager
drwxr-xr-x  6 root root   114 Sep 20 22:05 manager
drwxr-x---  5 root root    48 Sep 27 02:04 myApp
-rw-r-----  1 root root 42868540 Sep 27 02:06 myApp.war
drwxr-xr-x  3 root root    223 Sep 20 22:05 ROOT
root@158f955a62c6:/usr/local/tomcat/webapps#
```

