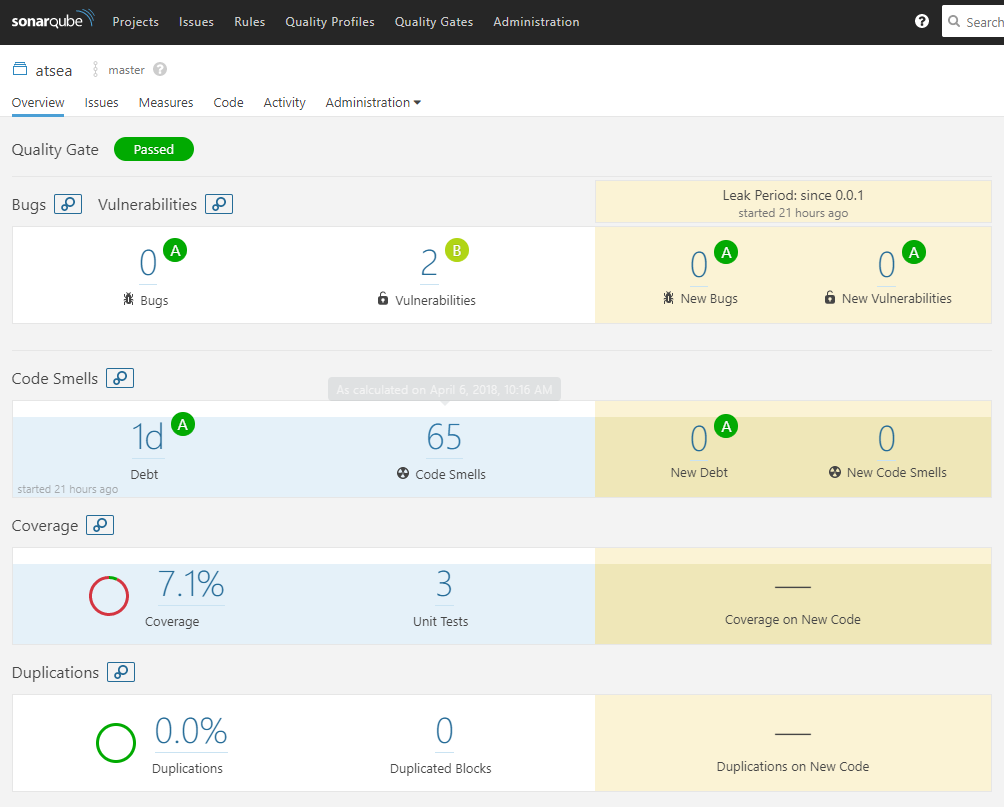
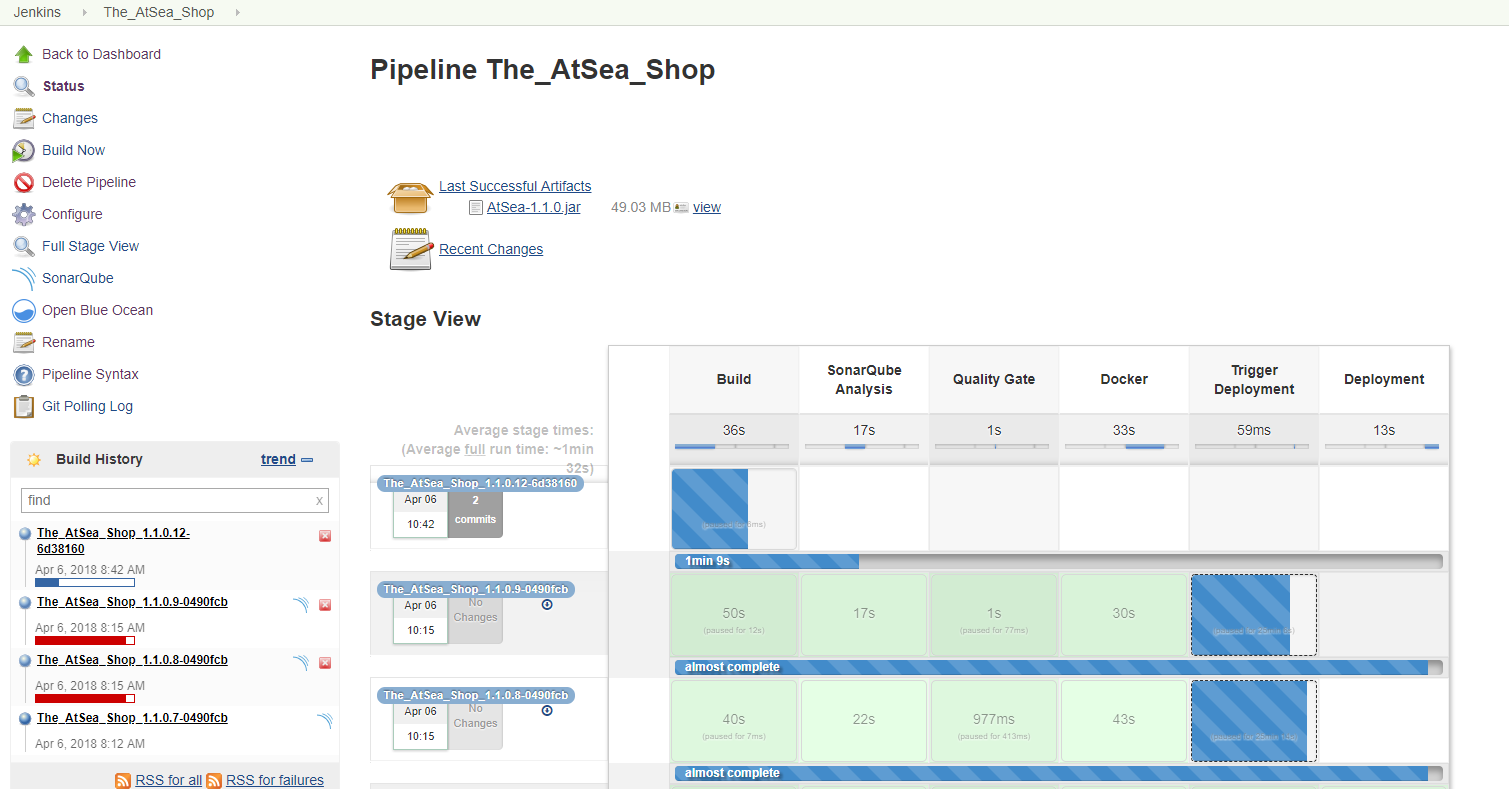
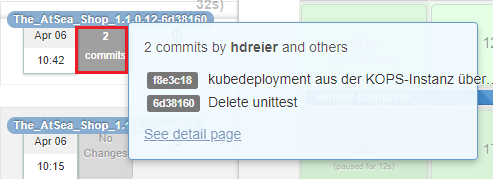
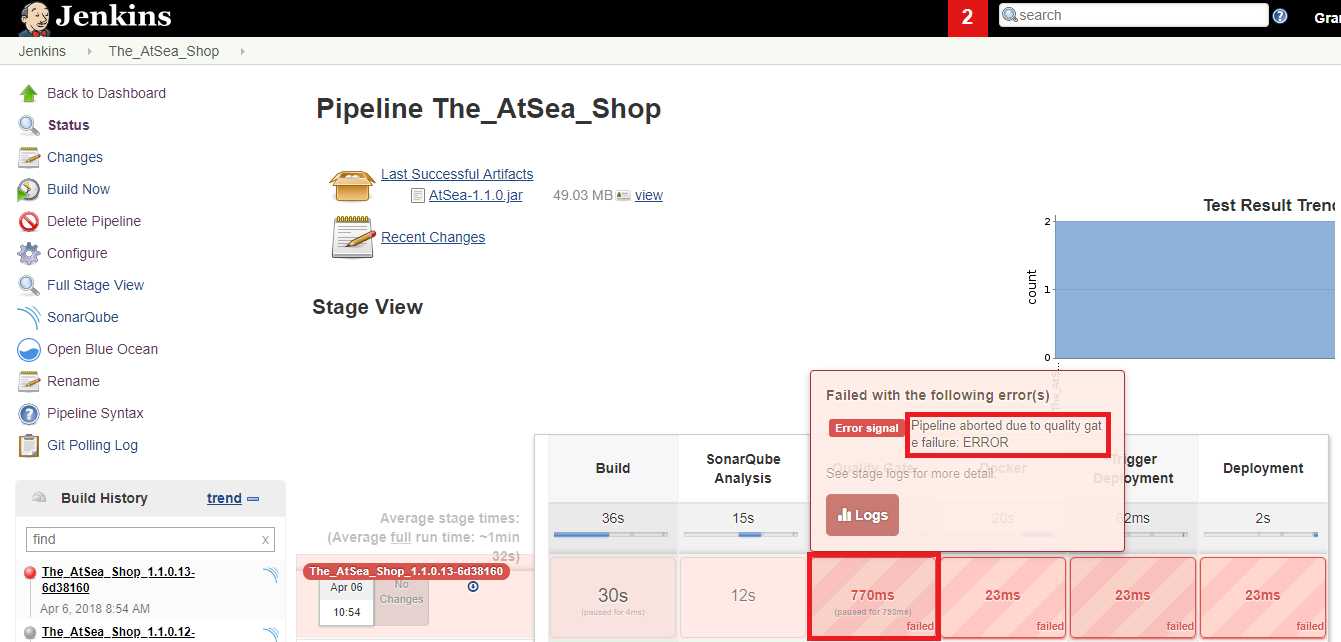
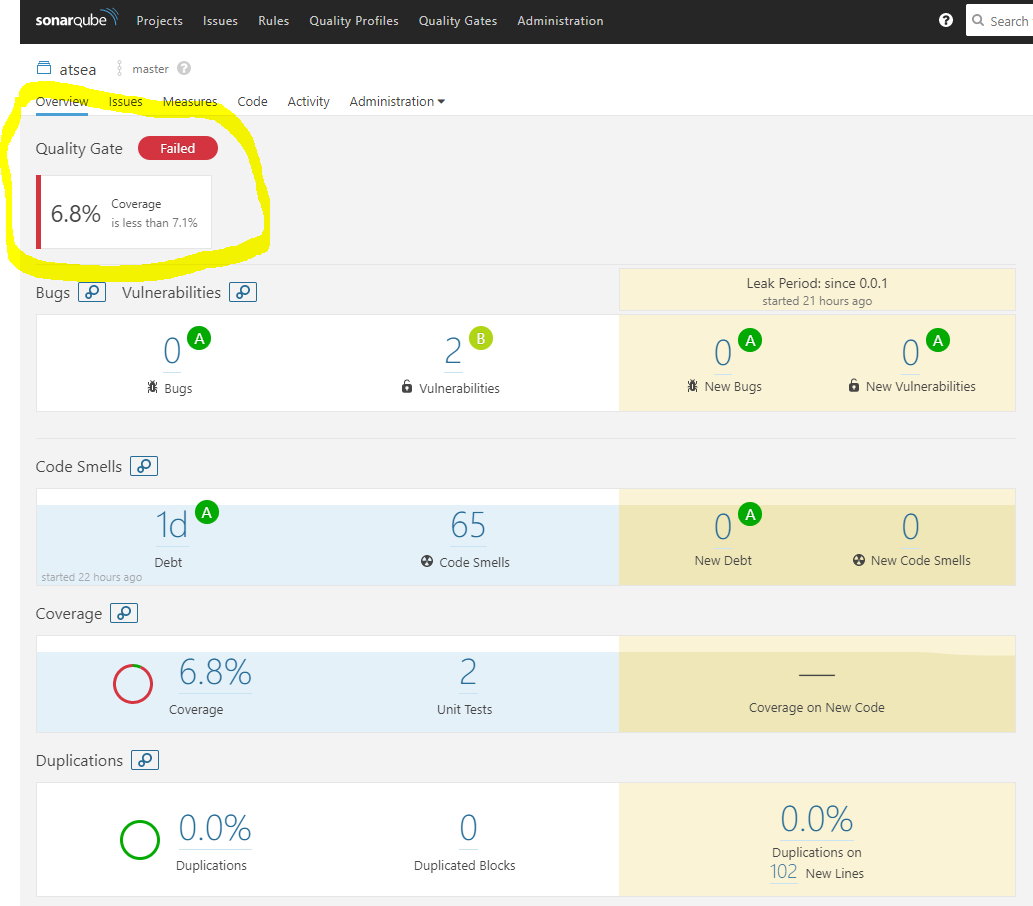
Automatisierung

# Schnellüberblick WebShop

# Java Code Anpassung mit SonarQube Rule Verletzung inkl. Git commit (Start der Pipeline)

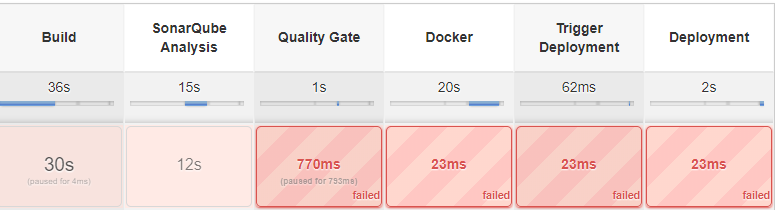
Grundlagen:

* Jenkins
  + admin:admin
  + ec2-18-194-223-76.eu-central-1.compute.amazonaws.com:8080
* GitHub
  + 368863656547:Materna.123
  + https://github.com/368863656547/368863656547.git
* Sonarqube
  + admin:admin
  + ec2-35-158-78-192.eu-central-1.compute.amazonaws.com:9000

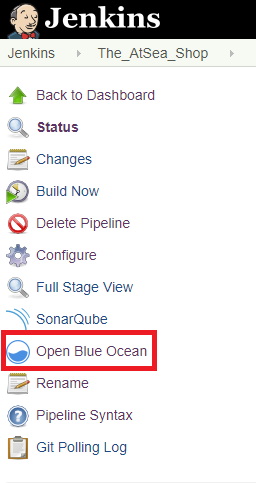
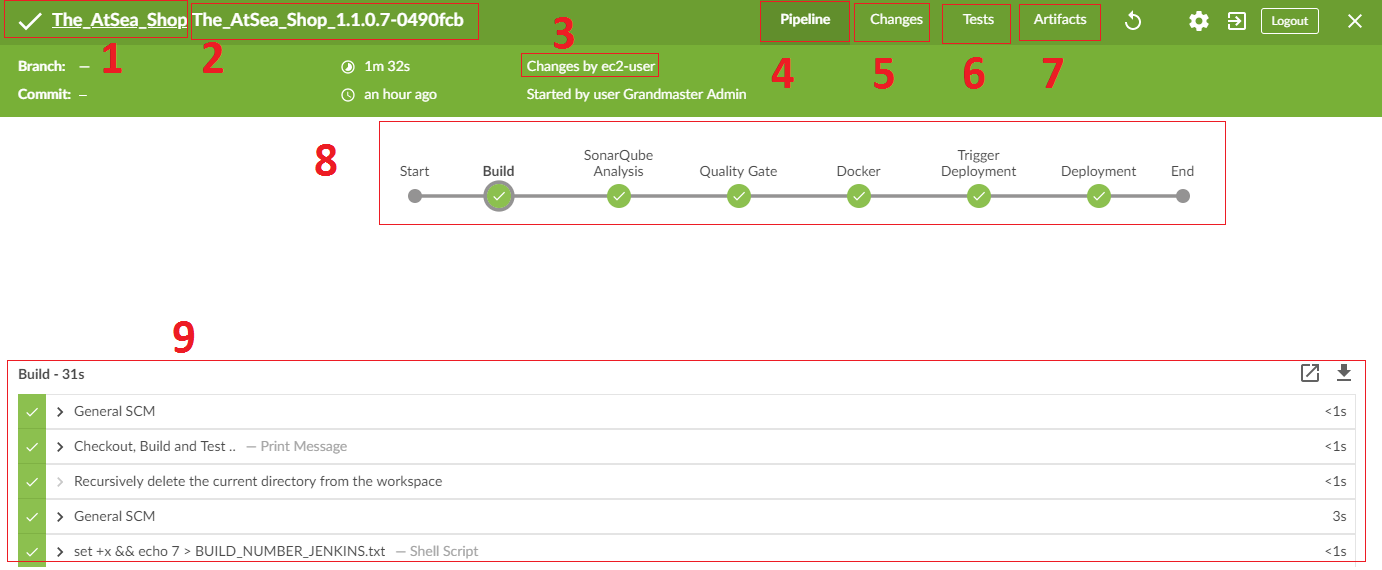
1. Aufruf Sonarqube
   1. Sonarqube aufrufen: ec2-35-158-78-192.eu-central-1.compute.amazonaws.com:9000
   2. Project „atsea“ aufrufen.
   3. Zeigen, dass dort keine Regeln verletzt sind.  
      
   4. Stand jetzt, es sind alle Sonar Anforderungen erreicht und das Quality Gate ist passed.
2. Jenkins aufmachen.
   1. Dort die Pipeline aufmachen und zeigen, dass alles gut ist  
      
3. Nun Commit nach GitHub.
   1. Unter „\app\app\src\test\java\com\docker\atsea\test\UnitTest.java“ einen Unittest auskommentieren.
   2. Änderung nach GitHub comitten.
   3. Nach dem Commit rennt der Jenkins in der Pipeline los und die App wird gebaut.
   4. Die Pipeline läuft los und mit Mouseover auf Commits, siehst du dann auch deinen Commit:  
      
   5. Da wir nun keine 7.1% CodeCoverage haben, ist das Quality Gate nicht erreicht und der Build knallt:
   6.   
      Wenn du mit der Maus auf die QualityGate Stage gehst dann öffnetsich das Fenster mit der Fehlermeldung und wir sehen, Quality Gate failure.
4. Wir wechseln zurück nach Sonarqube
   1. Das atsea Project sollte noch offen sein.
   2. Hier sehen wir nun auch, dass das QualityGate nicht erreicht wurde und dass die CodeCoverage Schuld ist:  
      
5. Showcase ist damit durchgelaufen.

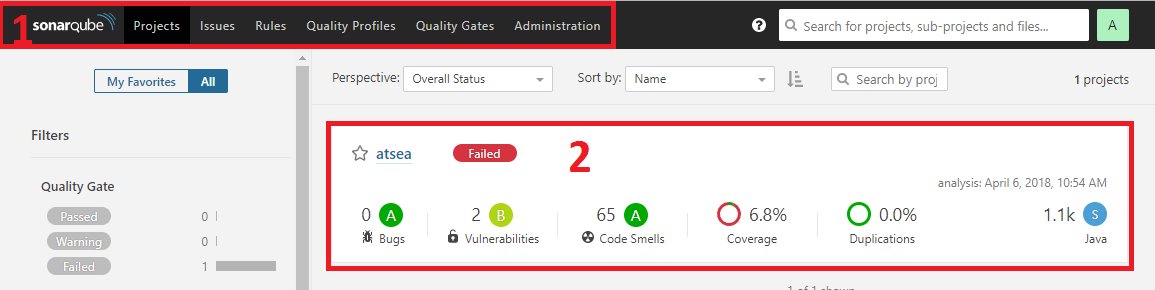
# CI/CD Pipeline zeigen (Quality Gates, Tests,

1. Jenkins Pipeline Classic
   1. Gehe auf den Jenkins



* + 1. Build = Build und Unittests
    2. SonarQube Analysis = Sonaranalyse
    3. Quality Gate = Check ob Quality Gate erreicht.
    4. Docker = Erstelle Docker Container, Tag ihn und Push ihn.
    5. Trigger Deployment= Willst du deployen ja oder nein? Mouseover und dann Promote!
    6. Deployment 0 Deployment gegen den Cluster.

1. Jenkins Pipeline Blue Ocean
   1. Erstmal auf den Jenkins gehen.
   2. Dann ins Projekt gehen.
   3. Im Projekt gibt es einen Button BlueOcean:  
      
   4. Nun landest du bei Blue Ocean:  
        
      1 = Projektname  
      2 = Buildname  
      3 = Changes from User  
      4 = Zeig mir die Pipeline  
      5 = Zeig mir die Changes  
      6 = Zeig mir die Tests  
      7 = Zeig mir die Artefakte des Builds (ich lasse die Jar hochladen)  
      8 = Auflisten der Pipeline Stages! Wenn du auf einen der Kreise/Stages klickst, wird dir …  
      9 = der Ablauf der Aufgaben der Stage angezeigt.
2. SonarQube
   1. Gehe nach Sonarqube



1 = Menü  
2 = Projekte

# Rule Verletzung beheben im Java Code, anschl. neuer Pipeline Lauf

* Einfach hier den Unittest wieder reinnehmen und einchecken.

# Manueller Trigger für PROD Deployment)

?????????????????????????????

# Prüfen des neuen Deployments (neue Applikation)

* Wenn du eine Änderung eincheckst dann wird diese deployt und du siehst dann die neue App nach dem Deployment.

Rollback

# Fehlerhaften Java Code einchecken und Deployment durchführen

* Einchecken usw. hast du ja bereits gemacht. Das Deployment läuft durch und die kaputte App wird deployt wenn:
  + Unittests noch durchlaufen
  + Quality Gate nicht failed
* Dann kannst du die kaputte App zeigen und fertig.

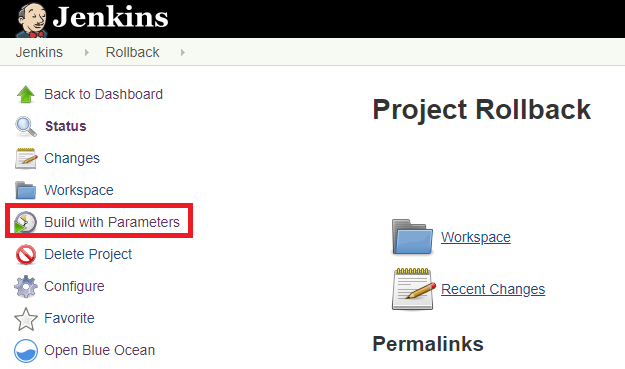
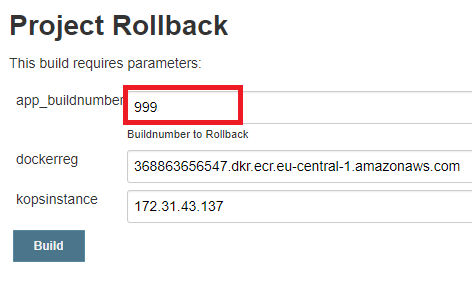
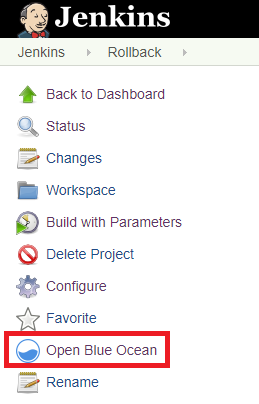
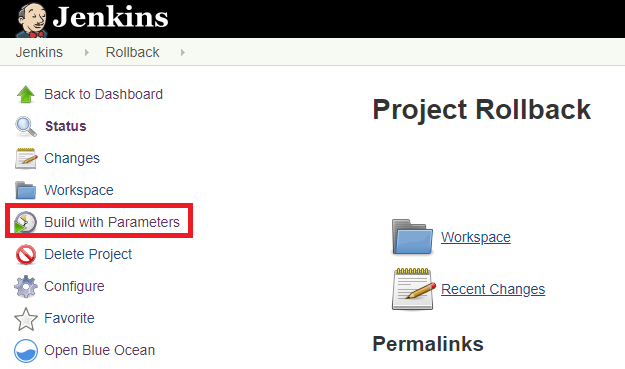
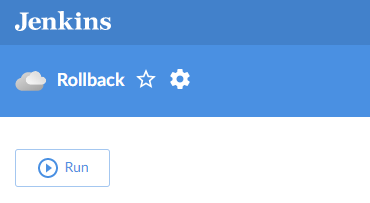
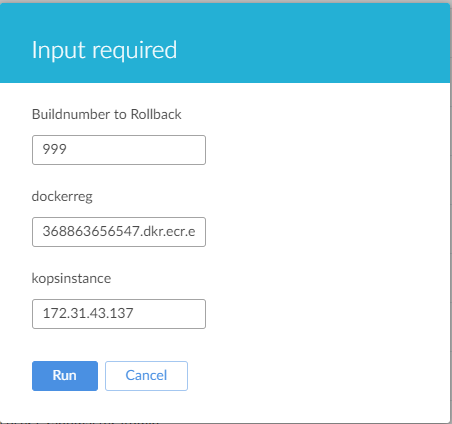
# Applikationsfehler über Web UI zeigen

* Zeige in der Oberfläche das alles kaputt ist.

# Daten für Post Mortem Analyse sichern

?????????????????????????????????

# Automatischen Rollback antriggern

* Zunächst musst du die letzte funktionierende Versionsnummer identifizieren
  + Wechsel nach AWS Amazon ECS
  + Dort appserver öffnen  
    <https://eu-central-1.console.aws.amazon.com/ecs/home?region=eu-central-1#/repositories/appserver#images;tagStatus=ALL>
  + Hier die letzte gesunde Version auslesen z.B. 1.1.0.7
* Auf den Jenkins wechseln
  + Rollbackprojekt auswählen
  + Jenkins Classic
    - Build with Parameters auswählen  
      
    - Hier die Buildnummer eintragen  
      
    - Auf Build hauen und der Rollback läuft.
  + Rollbackprojekt auswählen
  + Jenkins BlueOcean aufrufen  
    
    - Build with Parameters auswählen  
      
    - Auf Run Klicken  
      
    - Ein Fenster öffnet sich. Hier kann man dann die Versionsnummer auf die zurück gerollt werden soll, eingetragen werden. Dann auf Run und Rollback läuft:
    - 

# Webshop funktioniert wieder mit alter Version

* Ab auf die Seite und es sollte wieder gehen.