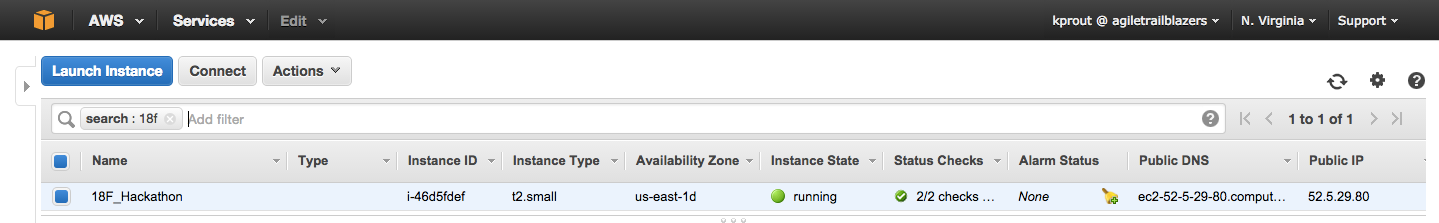
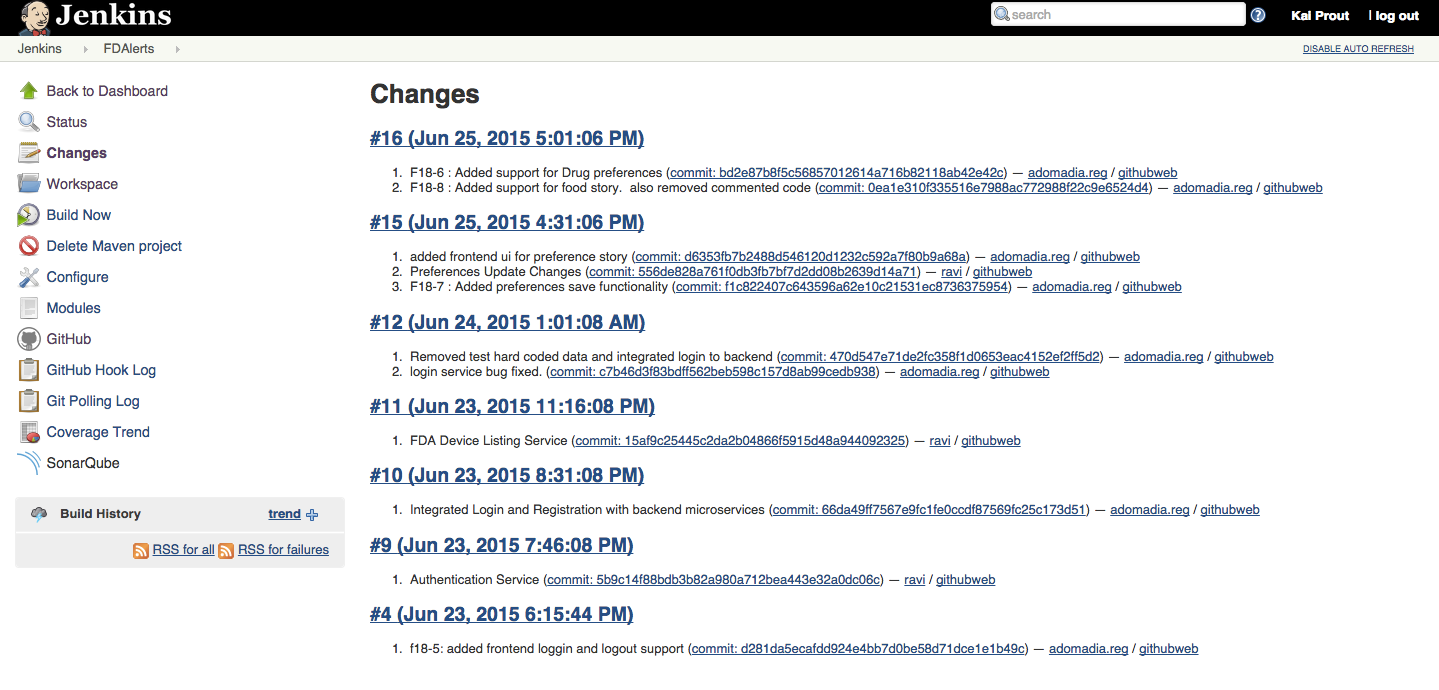
### Overview

The intent of this document is to demonstrate the DevOps processes executed during development of FDAlerts.

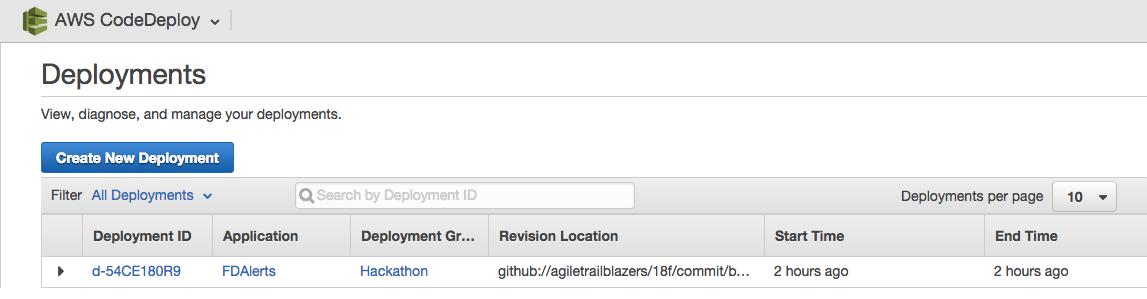
### Deployment

[FDAlerts.com](http://FDAlerts.com) is deployed on Amazon Web Services (AWS) which is a widely known Infrastructure as a Service tool. We chose a t2.small Elastic Computer instance to run our application. We used one of our custom AMI’s to spawn the instance with chef-client preinstalled to better streamline our DevOps process.

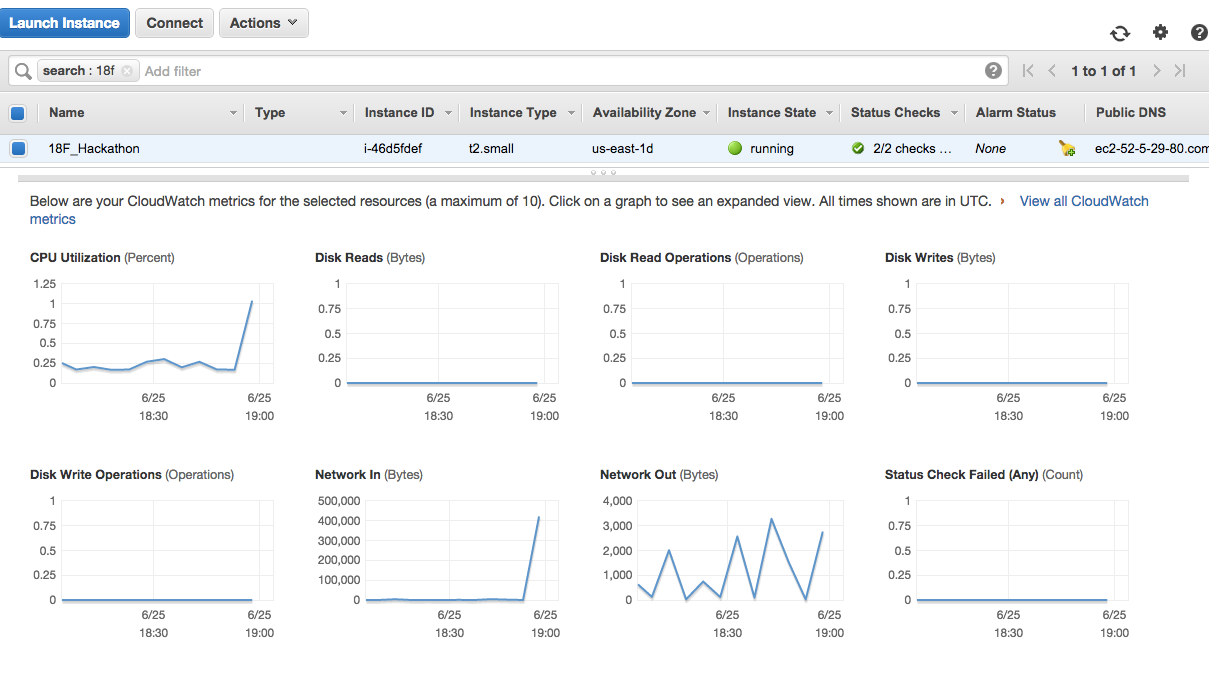
### Continous Integration

We used Jenkins, an open-source software application to monitor, build, and assist in deployment of [FDAlerts.com](http://FDAlerts.com). Our Jenkins project is linked to our Github repository build the master branch. Builds are triggered when changes are pushed to GitHub. When triggered, Maven finds the root pom.xml file and runs the goals set by Jenkins. If the build succeeds code is deployed to our EC2 instance through Amazon’s Code Deploy plugin, and SonarQube generates an analysis. If the build fails emails are sent to the appropriate members of the team and the build is not published.

### Continous Deployment

We are using Amazon’s Code Deploy to continuously deploy our application to our EC2 instance. Code will deploy after making merge or push to master and the Jenkins build is successful. We used Amazon’s Command Line Tools to achieve this functionality in combination with custom roles in Amazons Identity and Access Management service.

### Continous Monitoring

Hardware monitoring is done through Amazon’s CloudWatch. We are also using SonarQube for code coverage and vulnerability analysis.

### Configuration Management

In order to increase scalability efficiency, stability, and reduce costs we chose Chef as our configuration management tool. Our instance was configured by our Chef server as a client node with a role that allowed all the required dependencies to be installed and running on our EC2 instance.