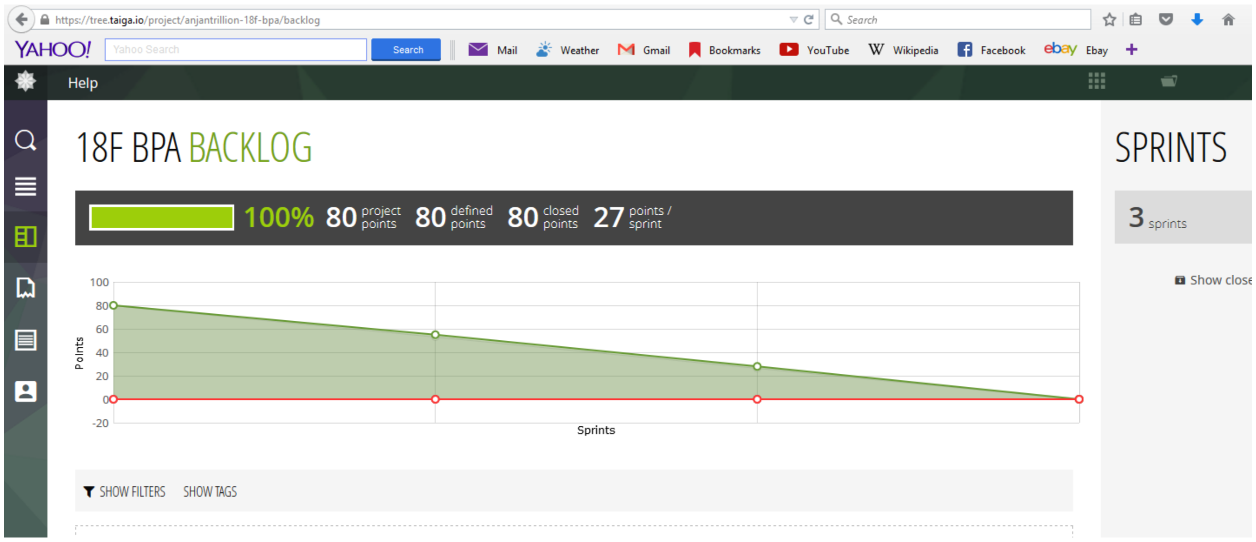
Trillion's Agile Software Development Approach for Design and Development

For RFQ 4QTFHS150004 Agile Delivery Services (ADS I)

Trillion has been using Agile development methodology since 2006. We have successfully implemented several small as well as large, turnkey and mission critical applications in commercial and government sectors. We leveraged this experience to create a compact set of Agile sprints for development a solution for the challenge posed by GSA 18F BPA. Our team comprised of members who were self-organizing and cross-functional and participated in flushing out user stories, design and development, unit and integration testing.

Sprint 0 comprised of understanding and detailing the requirements presented in the RFQ and creating a few skeleton user stories that created a bare bone structure of the prototype including wireframes, AWS setup and GitHub among others. Core development was during Sprint 1, Sprint 2 and Sprint 3, each lasting 5 days. User stories with acceptance were created after discussion among the team and with the product owner. Daily Scrum meetings provided everyone to come together formally to discuss work done the previous day and the plan for that day in addition to bringing up impediments if any. The last day of each sprint included sprint retrospective and review followed by sprint planning for the next sprint. A product backlog was created by the product owner from which user stories were chosen for each sprint which went into the sprint backlog.



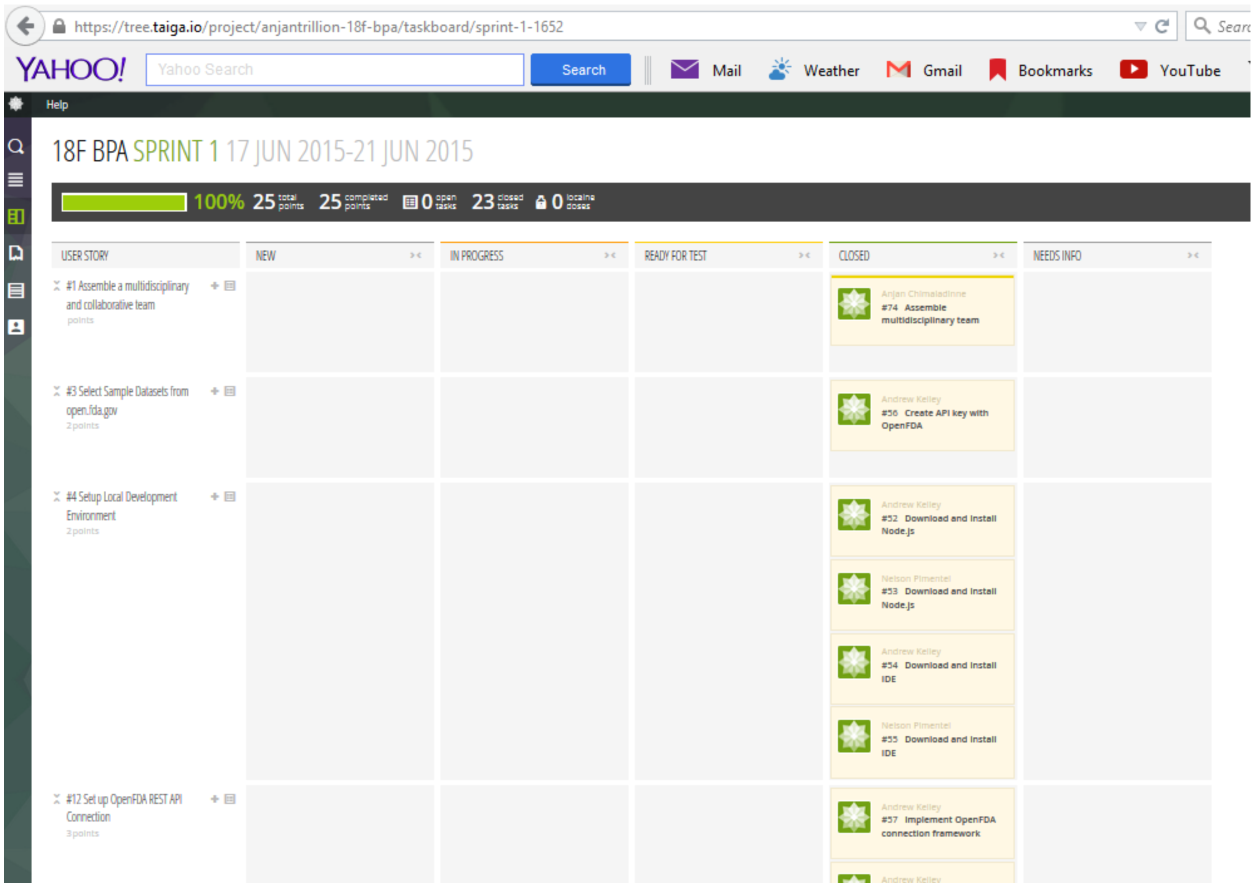
https://tree.taiga.io/project/anjantrillion-18f-bpa/backlog
Help
18F BPA BACKLOG
100% 80 project points 80 defined points 80 closed points 27 points / sprint
SPRINTS
3 sprints
Show close
Points
Sprints
SHOW FILTERS
SHOW TAGS

Figure 1: Release plan

An open source Agile Lifecycle Management (ALM) tool – Tiaga – was used for capturing all Agile artifacts. End of the sprint deployments were done on the Amazon instance and a demo was provided to the product owner with feedback taken that was loaded back into the product backlog. The product owner "resolved" the user stories to show acceptance of the deliverables.

Our release plan dates are

(1) Sprint 0: 6/17/2015
(2) Sprint 1: 6/17/2015 - 6/21/2015
(3) Sprint 2: 6/22/2015 - 6/26/2015
(4) Sprint 3: 6/27/2015 - 7/1/2015

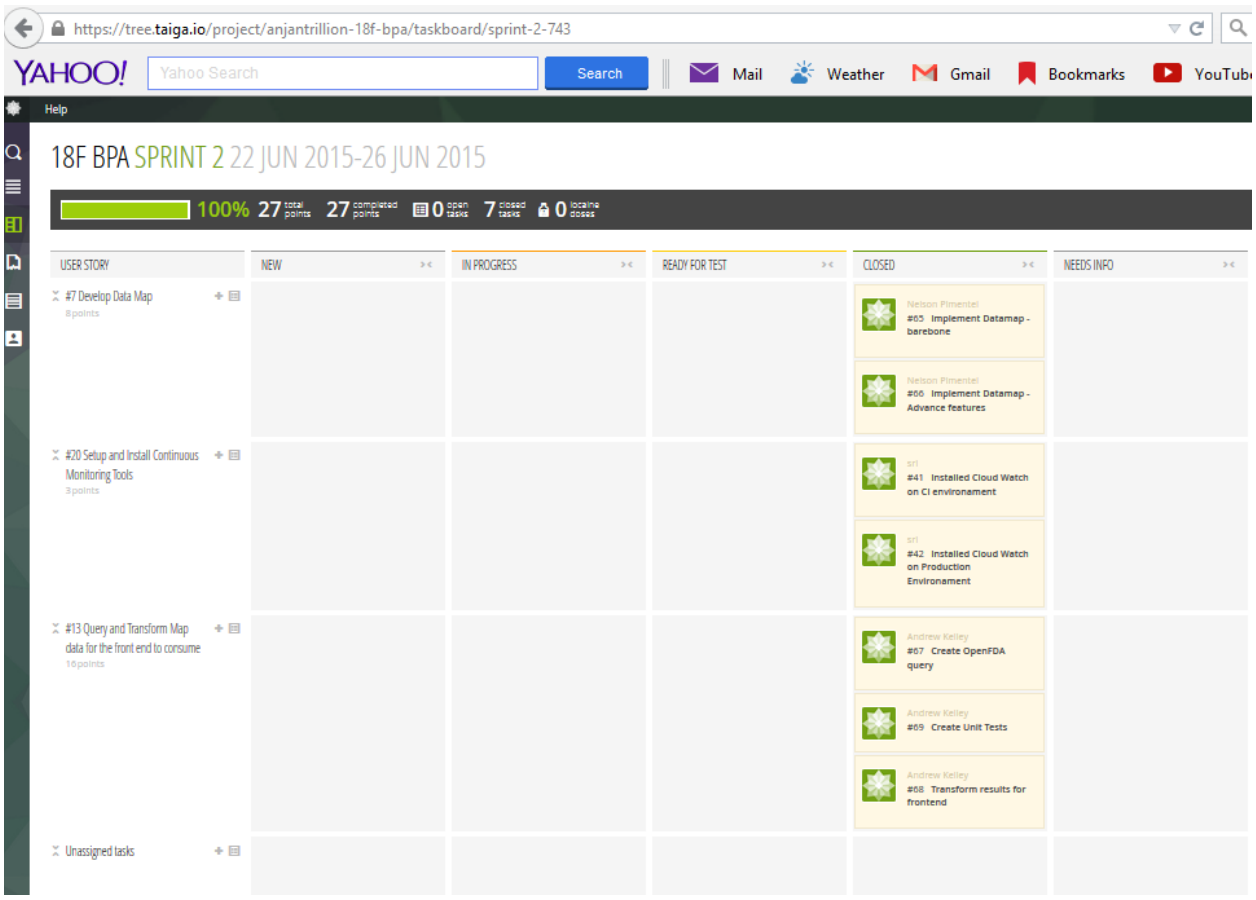

https://tree.taiga.io/project/anjantrillion-18f-bpa/taskboard/sprint-1-1652
YAHOO!
Yahoo Search
Search
Mail
Weather
Gmail
Bookmarks
YouTube
Help

18F BPA SPRINT 1 17 JUN 2015-21 JUN 2015

100% 25 total points 25 completed points 0 open tasks 23 closed tasks 0 iocaine doses

USER STORY	NEW	IN PROGRESS	READY FOR TEST	CLOSED	NEEDS INFO
#1 Assemble a multidisciplinary and collaborative team points				Anjan Chimaladinne #74 Assemble multidisciplinary team	
#3 Select Sample Datasets from open.fda.gov 2points				Andrew Kelley #56 Create API key with OpenFDA	
#4 Setup Local Development Environment 2points				Andrew Kelley #52 Download and Install Node.js Nelson Pimentel #53 Download and Install Node.js Andrew Kelley #54 Download and Install IDE Nelson Pimentel #55 Download and Install IDE	
#12 Set up OpenFDA REST API Connection 3points				Andrew Kelley #57 Implement OpenFDA connection framework Andrew Kelley	

Figure 2: Sprint 1

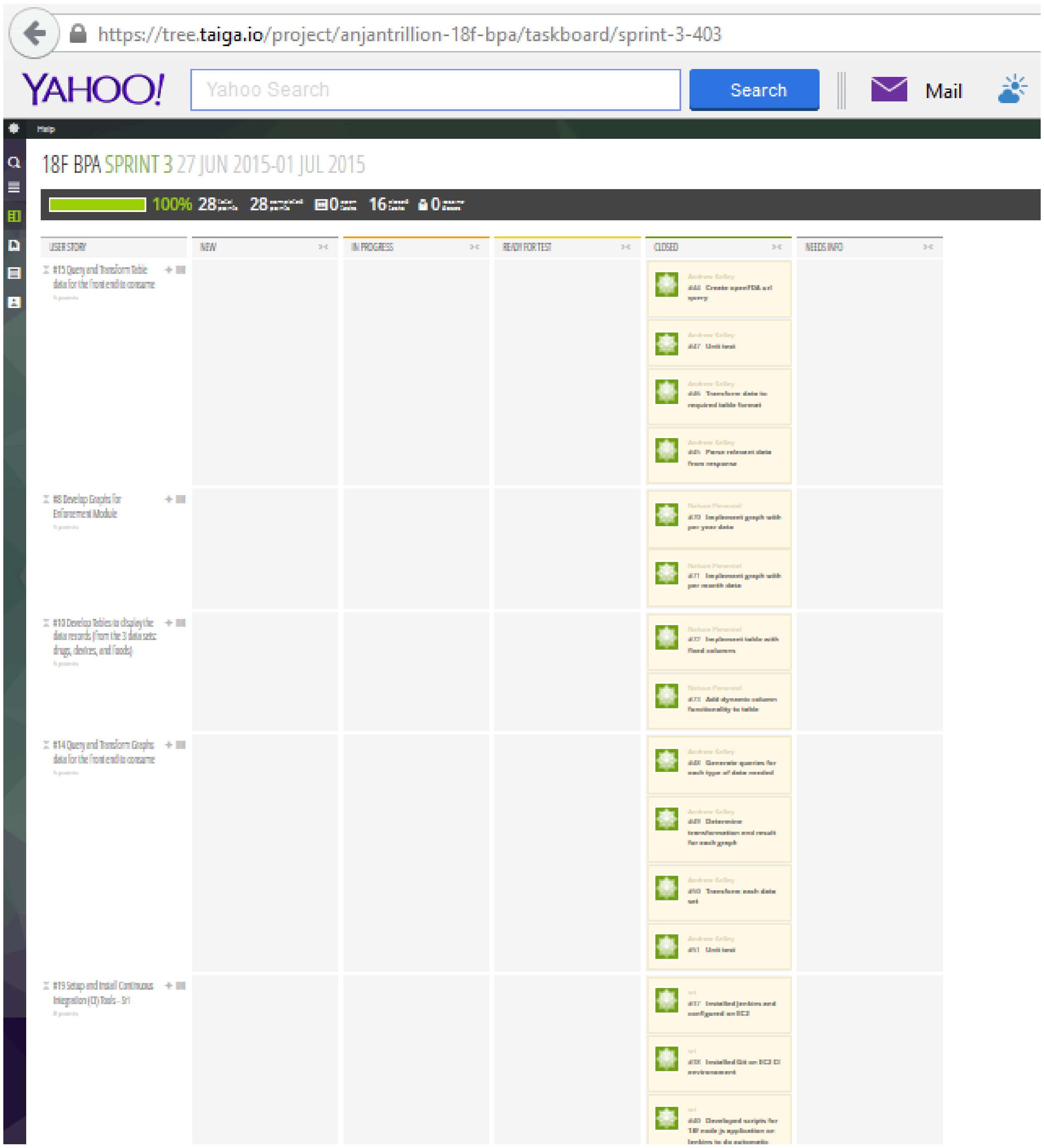

https://tree.taiga.io/project/anjantrillion-18f-bpa/taskboard/sprint-2-743
YAHOO!
Yahoo Search
Search
Mail
Weather
Gmail
Bookmarks
YouTub
Help

18F BPA SPRINT 2 22 JUN 2015-26 JUN 2015

100% 27 total points 27 completed points 0 open tasks 7 closed tasks 0 iocaine doses

USER STORY	NEW	IN PROGRESS	READY FOR TEST	CLOSED	NEEDS INFO
#7 Develop Data Map 8points				Nelson Pimentel #65 Implement Datamap - barebone Nelson Pimentel #66 Implement Datamap - Advance features	
#20 Setup and Install Continuous Monitoring Tools 3points				sri #41 Installed Cloud Watch on CI environament sri #42 Installed Cloud Watch on Production Environament	
#13 Query and Transform Map data for the front end to consume 16points				Andrew Kelley #67 Create OpenFDA query Andrew Kelley #69 Create Unit Tests Andrew Kelley #68 Transform results for frontend	
Unassigned tasks					

Figure 3: Sprint 2



https://tree.taiga.io/project/anjantrillion-18f-bpa/taskboard/sprint-3-403

18F BPA SPRINT 3 27 JUN 2015-01 JUL 2015

100% 28 28 0 16 0

USER STORY	NEW	IN PROGRESS	READY FOR TEST	CLOSED	NEEDS INFO
#15 Query and Transform Table data for the front end to consume				Andrew Kelley #44 Create openFDA url query	
				Andrew Kelley #47 Unit test	
				Andrew Kelley #46 Transform data to required table format	
				Andrew Kelley #45 Parse relevant data from response	
#8 Develop Graphs for Enforcement Module				Nelson Pimentel #70 Implement graph with per year data	
				Nelson Pimentel #71 Implement graph with per month data	
#10 Develop Tables to display the data records (from the 3 data sets: drugs, devices, and foods)				Nelson Pimentel #72 Implement table with fixed columns	
				Nelson Pimentel #73 Add dynamic column functionality to table	
#14 Query and Transform Graphs data for the front end to consume				Andrew Kelley #48 Generate queries for each type of data needed	
				Andrew Kelley #49 Determine transformation and result for each graph	
				Andrew Kelley #50 Transform each data set	
				Andrew Kelley #51 Unit test	
#13 Setup and Install Continuous Integration (CI) Tools - S1				sri #37 Installed Jenkins and configured on EC2	
				sri #39 Installed Git on EC2 CI environment	
				sri #40 Developed scripts for 18f node js application on Jenkins to do automatic	

Figure 4: Sprint 3

Trillion's Technical Approach for the Design and Development

The diagram below illustrates high-level architecture for Trillion's 18F prototype. The Business Tier is based on Trillion's Intelligence Management Platform (TIMP). The Business Tier houses the core backend logic for the system and exposes its functionality through REpresentational State Transfer (REST) based web services. TIMP consists on the Integration Tier component that has capabilities to invoke external services, databases, LDAP etc. The Presentation Tier provides a web based user interface that accesses TIMP using REST web services. For the purposes of the 18F prototype, no data tier components have been used. Typically, depending on the requirements,

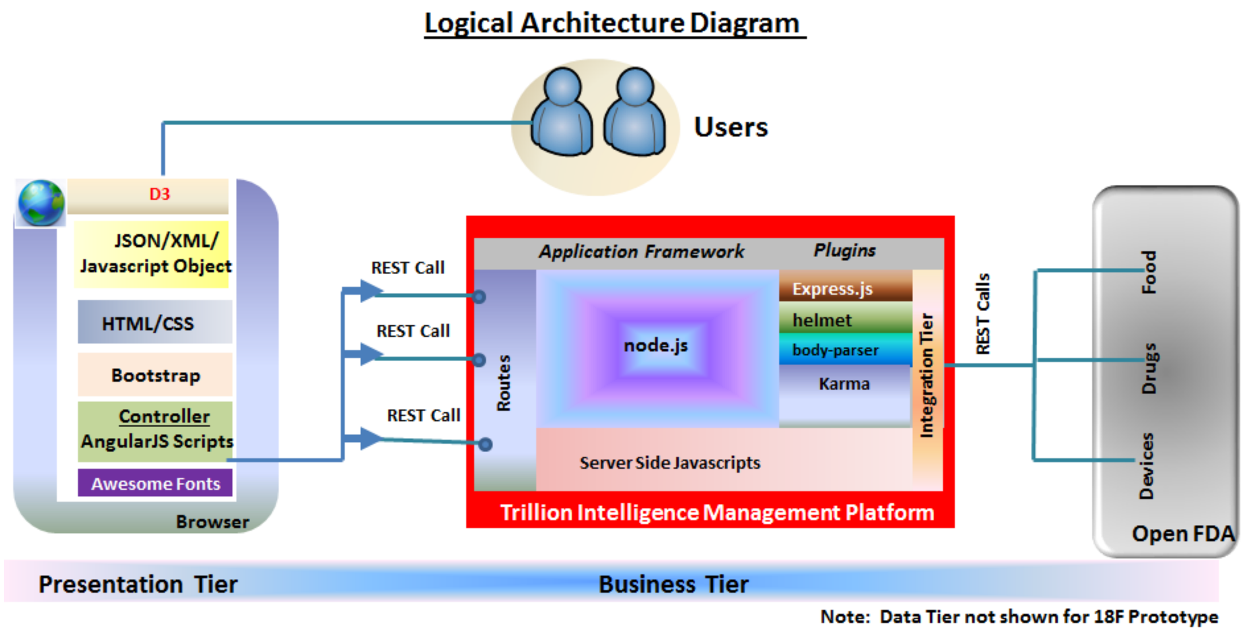


Figure 5: Logical Architecture of Trillion ODV Platform

TIMP provides easy integration with SQL and NoSQL based databases for a complete analytics solution.

Trillion's 18F prototype is implemented using lean, agile, open source technology stack. The Business Tier uses scaled down version of Trillion Intelligence Management Platform (TIMP) that is completely based on JavaScript technologies. Following is a list of Presentation tier and Business Tier technologies used:

Technology	Presentation Tier	Business Tier	Comments
Angular.js	x		Structural framework to build dynamic web pages
D3.js	x		D3 for Data-Driven Documents) is a JavaScript library for producing dynamic, interactive data visualizations in web browsers
Bootstrap	x		An HTML, CSS, JavaScript framework that you can use as basis for creating web sites or web applications

Technology	Presentation Tier	Business Tier	Comments
Awesome Fonts	x		Font and CSS toolkit
Node.js		x	Node.js is an open source, cross-platform runtime environment for server-side and networking applications. Node.js applications are written in JavaScript.
Express.JS		x	ExpressJS is a framework of Node.js that allows one to use several very useful and powerful features without having to reinvent the wheel, helps organize application's routing and use any template based solution with minimal effort.
Forever		x	Allows a script to be run continuously.
Helmet		x	Helps lock down and secure our web applications.
Body-Parser		x	Node.js body parsing middleware
Compression		x	Helps compress data exchanged between different tiers.
glob		x	File pattern matching package
Request		x	HTTP request client
Karma		x	Test driven development
Jasmine		x	package that contains helper code for developing and running tests for node-based projects

Express.JS uses the concept of 'Routes'. Routing refers to determining how an application responds to a client request to a particular endpoint or URI. Each route can have one or more handler functions, which is/are executed when the route is matched. TIMP exposes endpoints to request data for Open FDA's Device, Food and Drug related data. The handler functions invoke appropriate webservices exposed by Open FDA, process the data and send the response to requestor.

The Presentation Tier utilizes the Model-View-Controller pattern based on the Angular.JS framework. Since there were no Authentication/Authorization requirements, Trillion's 18F prototype has disabled this logic but these components can be easily enabled as and when required. Figure 6 below illustrates the development/deployment diagram for Trillion's 18F prototype.

TRILLION TECHNOLOGY SOLUTIONS

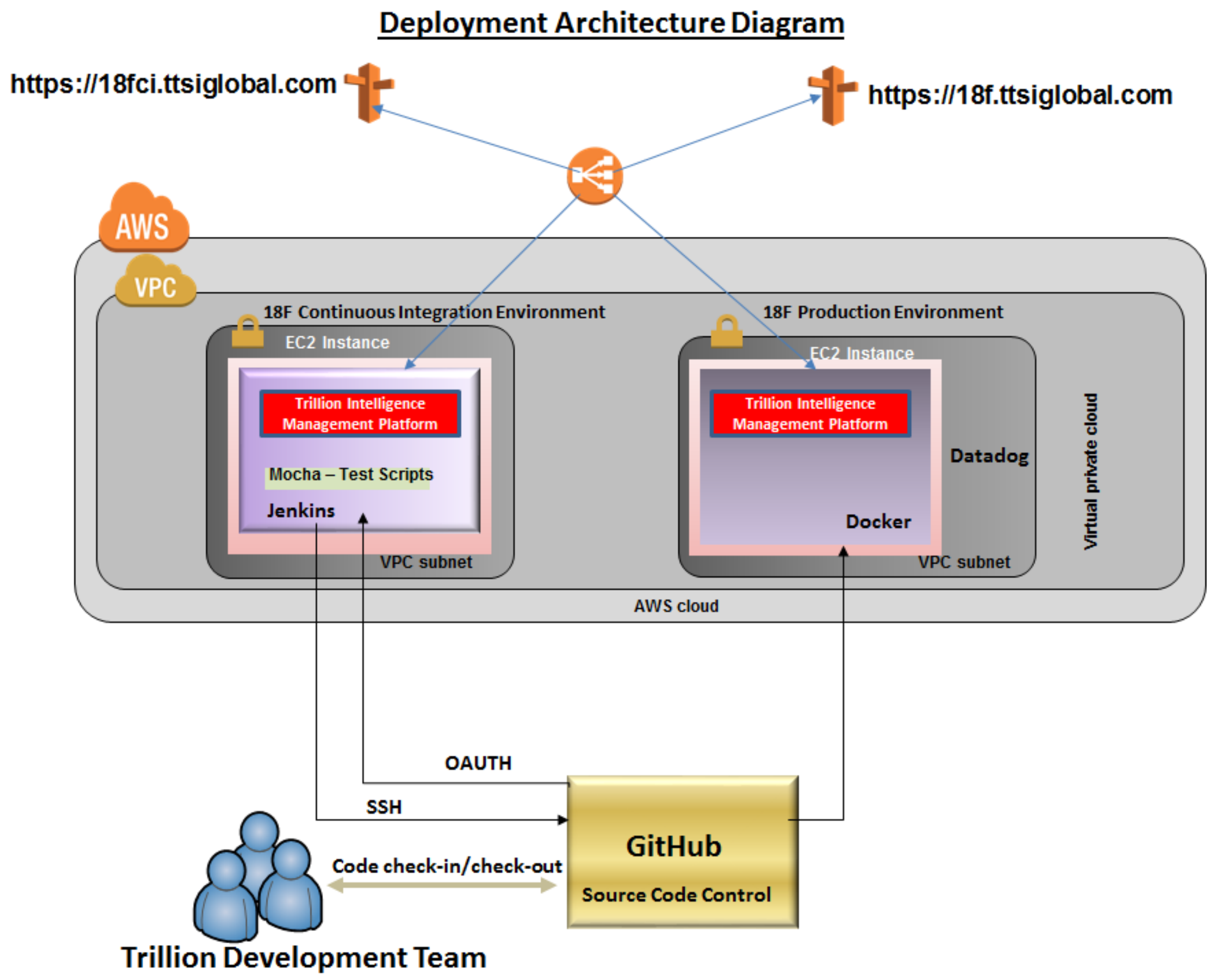

Deployment Architecture Diagram
https://18fci.ttsiglobal.com
https://18f.ttsiglobal.com
AWS
VPC
18F Continuous Integration Environment
EC2 Instance
Trillion Intelligence Management Platform
Mocha – Test Scripts
Jenkins
VPC subnet
18F Production Environment
EC2 Instance
Trillion Intelligence Management Platform
Datadog
Docker
VPC subnet
Virtual private cloud
AWS cloud
OAUTH
SSH
Code check-in/check-out
GitHub
Source Code Control
Trillion Development Team

Figure 6: Infrastructure and Deployment of Trillion ODV Platform

GitHub is used as the Source code repository. Two Amazon Web Service EC2 instances, one serving as the continuous integration and test server and the other serving as the Production server, are deployed in separate subnets to firewall Production and test environments. The continuous integration server, using Jenkins, checks out the latest code on a continuous basis, builds it and deploys the code. Mocha test framework is utilized for running automated test scripts.

The Production Server utilizes Docker container. Docker allows an application to be packaged with all of its dependencies into a standardized unit for software development. Docker containers wrap up a piece of software in a complete file system that contains everything it needs to run: code, runtime, system tools, and system libraries – anything you can install on a server. This guarantees that it will always run the same, regardless of the environment it is running in. The designated stable version of the code from GitHub is used for in the Production server.

Datadog provides continuous monitoring of Trillion ODV Platform in production.