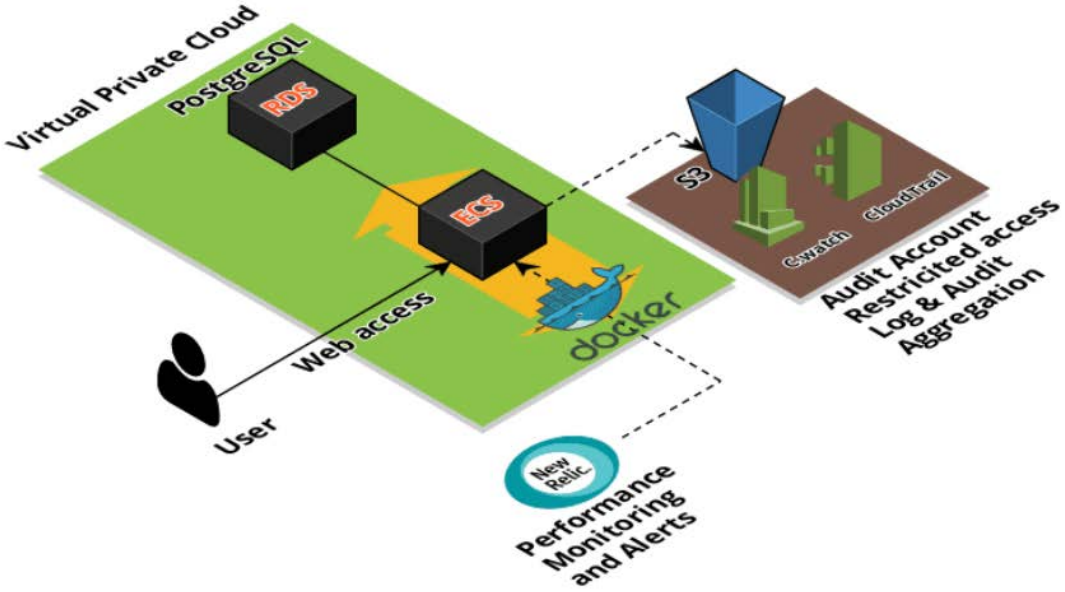
Infiniti KMT Application Architecture

Overview

This application architecture document includes the KMT prototype application architecture, Continuous Integration/Continuous Deployment (CI/CD) pipeline and shows the code flow from client UI, to JavaScript library, to REST service to database, pointing to code in the GitHub repository. This document is a supplement to the Technical Approach document that is stored README.md in Github: https://github.com/infiniticg/Infiniti-KMT

Infiniti KMT Prototype Application Architecture



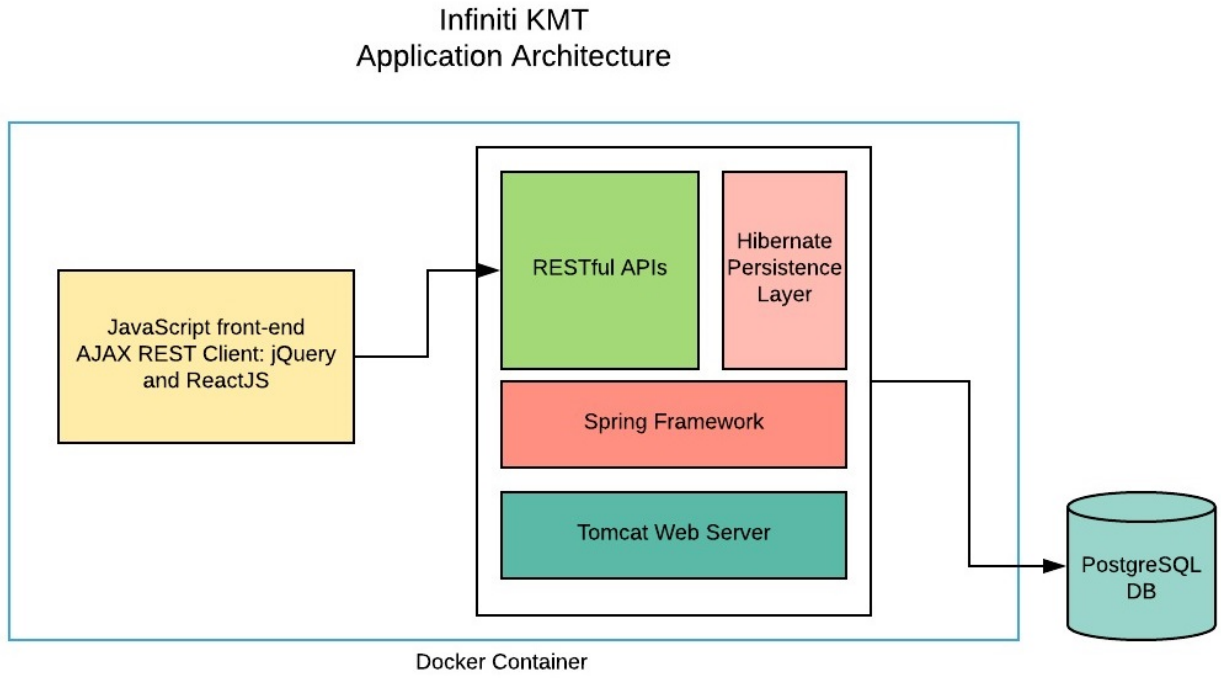
The application architecture diagram shows the major components in the Infiniti KMT prototype environment which is a AWS IaaS and PaaS environment. Included in this architecture are:

- AWS Elastic Container Service (ECS) - ECS is a highly scalable, high-performance container orchestration service that supports Docker containers and allows you to easily run and scale containerized applications on AWS
- Amazon Elastic Container Registry (ECR) - ECR is a fully-managed Docker container registry that makes it easy for developers to store, manage, and deploy Docker container images. ECR is integrated with AWS ECS, simplifying the development to production workflow. Amazon ECR eliminates the need to operate your own container repositories or worry about scaling the underlying infrastructure.
- AWS Relational Database Service (RDS) - RDS, as a PaaS, makes it easy to set up, operate, and scale a relational database in the cloud. RDS provides cost-efficient and

resizable capacity while automating time-consuming administration tasks such as hardware provisioning, database setup, patching and backups. For the KMT prototype we chose RDS Postgres which is an open source database.

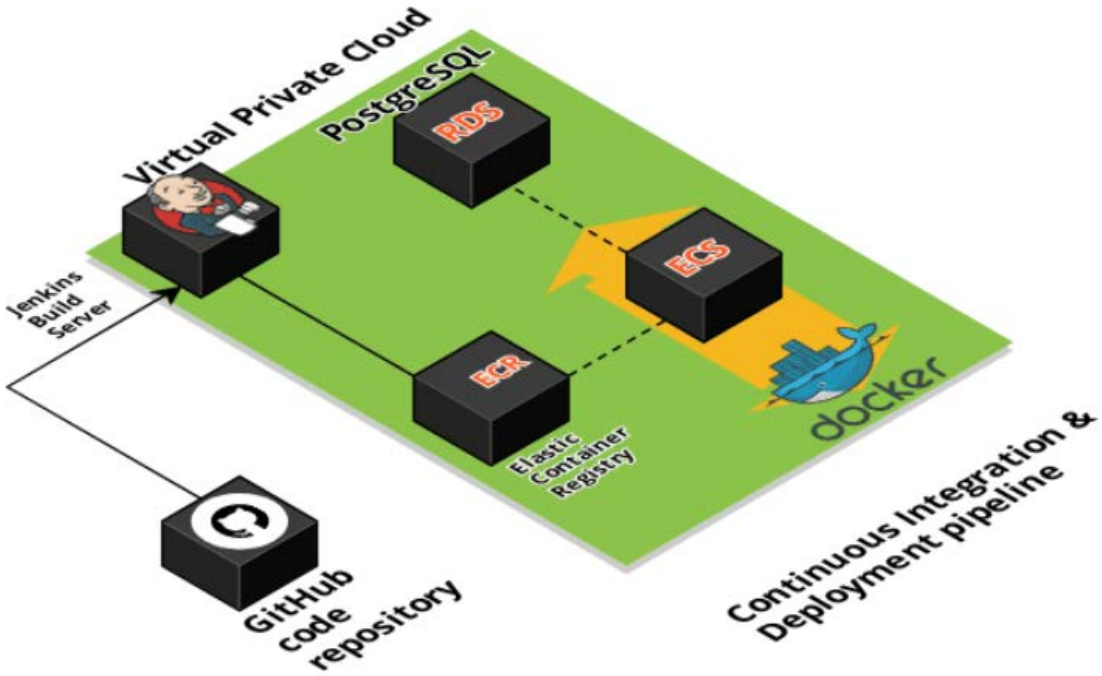
- The Infiniti KMT prototype runs inside a Docker container which is running in ECS and connecting to the Postgres database
- New Relic Agent is deployed in AWS ECS to monitor application health and performance. Infiniti is a partner with NewRelic and has implemented NewRelic in many large environments.
- Audit trail logs are saved into an AWS S3 bucket in a master account, away from the deployment account, for safe keeping.
- Resilience is provided by an auto-scaling group managed by ECS and a Multi-AZ DB Deployment that is also backed up regularly.

The diagram below shows the code flow from client UI, to JavaScript library, to REST service to database, pointing to code in the GitHub repository.



CI/CD Pipeline:

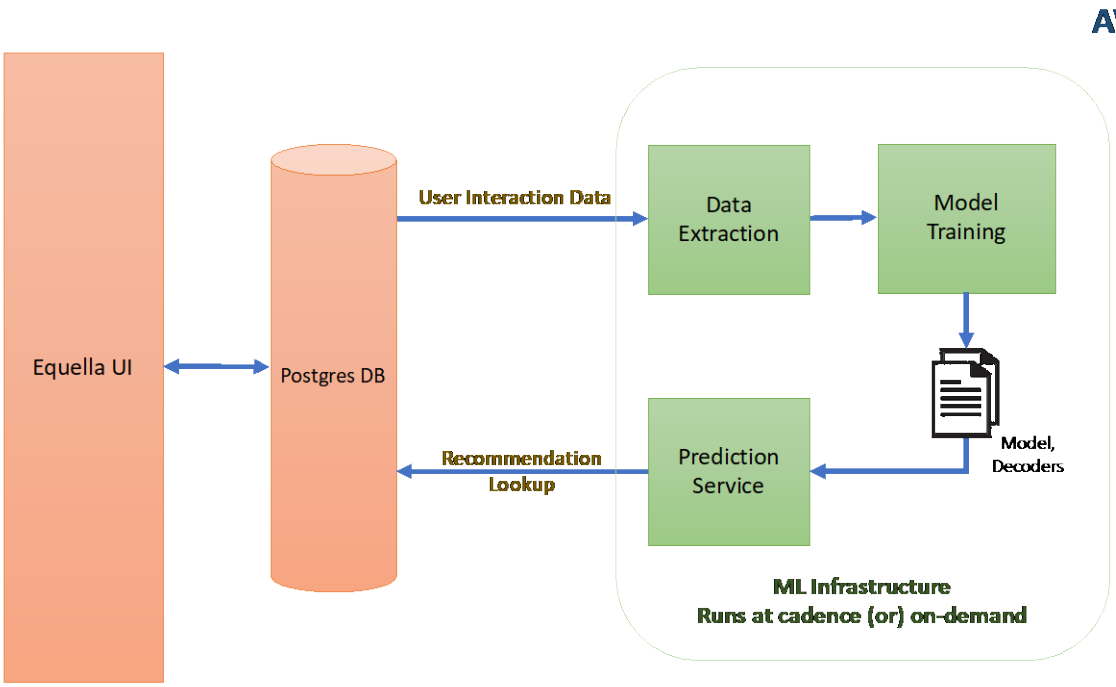
The diagram below shows the Continuous Integration/Continuous Development (CI/CD) pipeline created and used by the Infiniti team for the KMT prototype project.



- When a build is triggered, Jenkins downloads the source files from the GitHub repository.
- Tests are automatically run
- When the build completes the resulting artifact is a Docker Image.
- This Docker image is pushed to AWS ECR (Container Registry) and later pushed to AWS ECS (Elastic Container Service) for Deployment

Machine Learning and Predictive Analytics Architecture

ML/PA was integrated into the Infiniti Knowledge Management solution deployed in AWS. Outlined below is a high-level architectural view of how the implementation looks in production.



Machine Learning and Predictive Analytic Tools

The following table summarizes the tools utilized in the project.

Vector	Tools
Infrastructure	AWS: This include EC2-instances, S3 Storage to support extract from Postgres, ML back-bone.
Language	Python 2.7
Packages (Python)	**Scikit-learn, Pandas:** These are robust open-source Python packages for data science and machine learning. **Surprise:** An open-source recommender system building package. These packages are actively developed and are built and optimized for speed and efficiency. Using open-source solutions allows us to do rapid prototyping and development while keeping costs low.