# Abstract

The purpose of this document is to demonstrate how developers can create a meaningful business application using the most agile and developer friendly stack for creating enterprise class, production ready applications. The full development stack is referred to as ANEML (Angular, Node.js, Express, MarkLogic) and is pronounced “***Animal***”. Using a clever moniker, such as ANEML, can be helpful to wining hearts and minds of the development community, similar to the MEAN stack (see <http://mean.io>)

The demonstration application is a simple application called TweetDeck. This application allows a user to search for tweets stored in a MarkLogic database and plot the geographic location of the Tweeter on Google Maps.

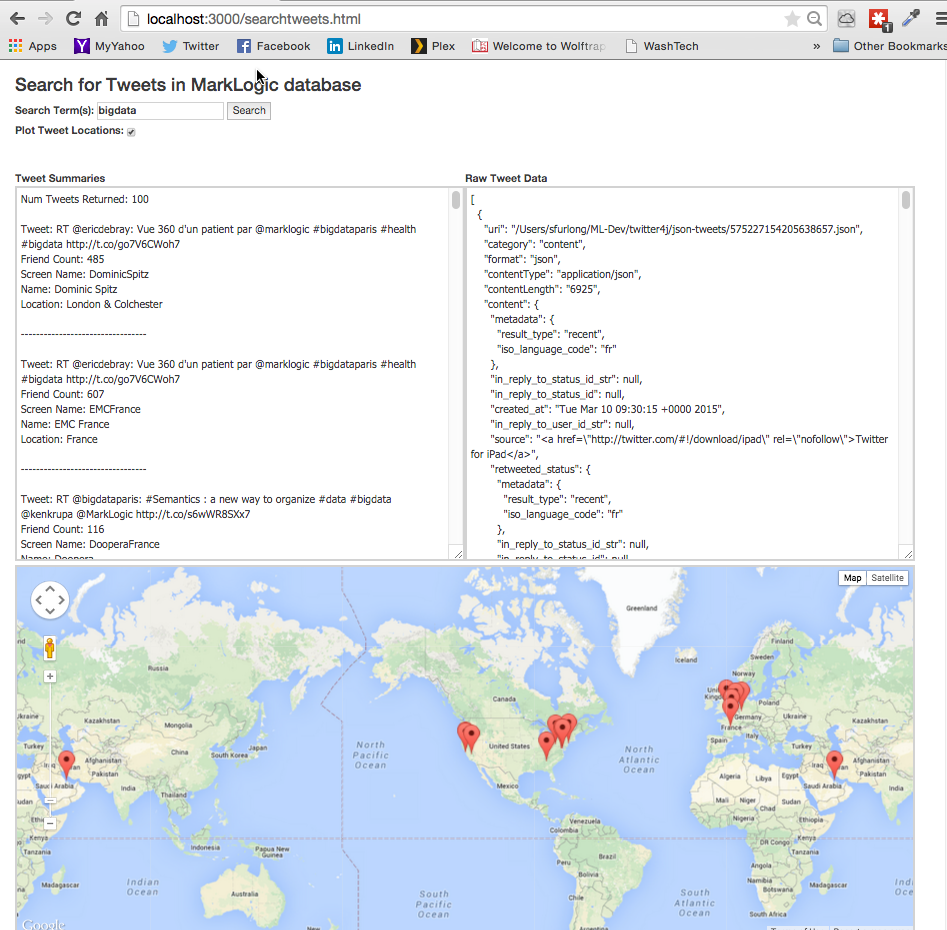


Figure TweetDeck a Sample ANEML Application

The rest of this document describes a step-by-step guide on how to recreate this sample application using the ANEML stack.

I developed this application as a way to learn the MarkLogic 8 platform features after joining the company. I hope that others new to the MarkLogic 8 platform find this project useful. For further information, assistance, or comment, please contact Steve Furlong at [steve.furlong@marklogic.com](mailto:steve.furlong@marklogic.com).

# Architecture Overview

There are two components:

1. Node.js Server. This code is in ./myapp/furserver.js

To start the server type: "node furserver.js"

It will now be listening at: 3000

Port is specified in furserver.js

Node API Docs: http://developer.marklogic.com/learn/node-client-api

2. Angular JS Client. This code is in ./index.html and ./script.js

3. Install MarkLogic Node.js development client

sudo npm install marklogic

\*See detailed instructions in the ML node-dev.pdf Developers Documentation



# ANEML Stack Installation

This section provides a step-by-step guide for installing the minimal components to develop using the ANEML stack. Only strict minimum components are used to provide the developer with a solid understanding of what layers of the stack contribute to specific architectural capabilities.

## Install Node.js

* Download install binaries from <https://nodejs.org>
* Once you've installed node, you will have an npm command (node package manager).
* Create a package.json file in the directory of interest, if it does not exist already, with the npm init command.
  + $ npm init

## Install Express

* Install Express in the app directory
  + $ npm install express –save
  + $ npm install express

## Install MarkLogic Node.js Client

* $ npm install marklogic –save

## Install Bootstrap CSS Style Sheets (OPTIONAL)

* $ npm install bootstrap

## Install MarkLogic and Database Setup

# TweekDeck Application Code Deployment

## Install Demo Code from github

* get from GitHub

# To Run The Demo

## Start the Node.js Server

* $ node server.js

# Testing

CTS:Search in Query Console

// find all documents with the word "car" and count them

var count = 0;

var results = new Array();

for (var result of cts.search(cts.andQuery([

cts.wordQuery("marklogic")])) ) {

count++;

//results.push(result);

};

results.push(fn.concat("Count = ", count));

results;

# Appendix

## Installing & Uninstalling MarkLogic Database

* Install: $ rpm –i <<marklogic-rpm-name>>
* Start: $ /etc/init.d/MarkLogic start
* Stop: $ /etc/init.d/MarkLogic stop
* Uninstall: $ rpm –e MarkLogic
* Remove DB Files: $ rm –fr /var/opt/MarkLogic

## Working with Git

1. git init
2. vi .gitignore
   1. Add node\_modules
3. Git add -A //Recursibly adds all files to local git repository
4. Git commit –m “<<comment>>” //local commit
5. Git remote add origin <https://sfurlong@github.com/sfurlong/aneml.git>
6. git push -u origin master
7. Git status
8. Git log
9. Git diff head
10. Git push

## Future Enhancement Opportunities

1. Query with joins between multiple document types
2. Add pagination to the UI when > 100 tweets are returned
3. AuthN via LDAP
4. AuthZ based on user Role
5. Reporting
   1. Tableau
   2. Pentaho
   3. Qlick
6. Semantics and ontologies