## Challenge Name: Build an #Awesome Demo with Amazon DynamoDB

Challenge Details

<http://www.cloudspokes.com/challenges/1362>

Solution Submitter ID

romin

Solution Submitted Profile URL

<http://www.cloudspokes.com/profile.html?username=romin>

Solution File Attached

**amazondynamodb\_romin.zip**

**Overview**

The ZIP file contains:

* This document
* **Amazondynamo-Java-Project.zip** : An Eclipse project. You will need to build it against the latest Amazon AWS SDK for Java 1.3.1
* **Amazondynamo-Node-Project.zip** : A NodeJS Project.

**Description of Solution**

The solution that is presented here is a API for various Data Sets for the city of Mumbai. I decided to take a different approach and provide an API instead of creating a bespoke solution.

The high level idea is that various Mumbai Data Sets will be stored in Amazon Dynamo and they will be provided in the form of a REST API to applications that will require them. The vision is that overtime more datasets will be added and clients can easily check the versions and update their data bases.

The datasets that are currently provided are:

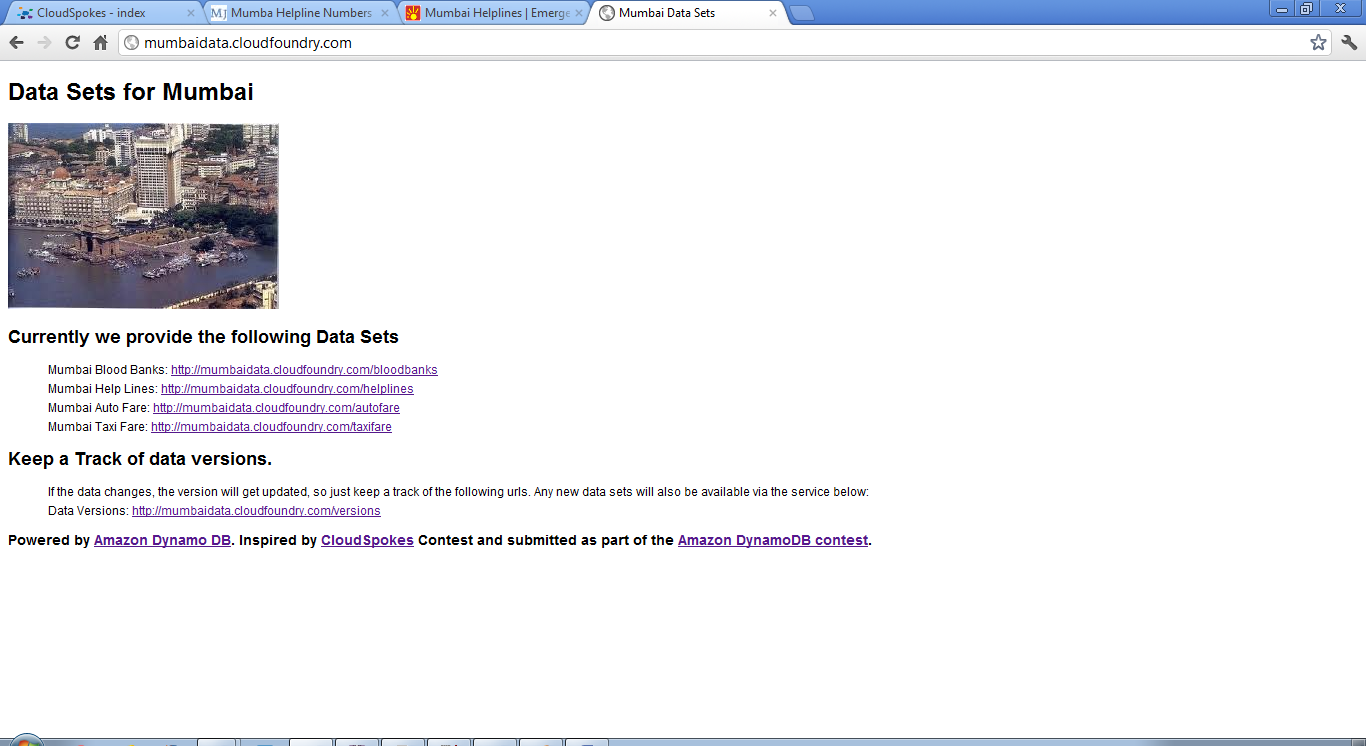
1. Mumbai Help Line Phone Numbers
   1. Various phone numbers for Emergency Services, Police, Local Municipalities, etc.
2. Mumbai Blood Banks Directory
   1. List of nearly 60 Blood Banks in the city of Mumbai with their area names and phone numbers.
3. Mumbai Taxi Fare Rate Card
   1. Current Rate card for the Mumbai Taxi. Simply take the Auto Meter Reading and it will give you the current fare. The current rate card published by the Mumbai Administration is provided. As the rate card changes, the new updates will be incorporated and clients can easily get the latest rate card.
4. Mumbai Auto Fare Rate Card
   1. Current Rate card for the Mumbai Autorickshaw. Simply take the Auto Meter Reading and it will give you the current fare. The current rate card published by the Mumbai Administration is provided. As the rate card changes, the new updates will be incorporated and clients can easily get the latest rate card.

All one has to do is access the data is to invoke the REST API endpoints for the dataset. The application (**a NodeJS solution talking to Amazon DynamoDB**) is hosted solution at CloudFoundry PaaS.

The solution can be accessed via the browser at:

<http://mumbaidata.cloudfoundry.com>

If you visit the URL, you will see the home page as shown below:



You will see the end points for the various services as listed below:

* Data Versions: <http://mumbaidata.cloudfoundry.com/versions>
* Mumbai Blood Banks: <http://mumbaidata.cloudfoundry.com/bloodbanks>
* Mumbai Help Lines: <http://mumbaidata.cloudfoundry.com/helplines>
* Mumbai Auto Fare: <http://mumbaidata.cloudfoundry.com/autofare>
* Mumbai Taxi Fare: <http://mumbaidata.cloudfoundry.com/taxifare>

The Data Set returned is in JSON format.

**A Note on DataSet Updates:**

The first service endpoint i.e. **/versions** is a metadata about the current dataset versions. All clients can use this endpoint to see the new or current datasets and get their current versions. If the version is new, they can simply get the new dataset as needed.

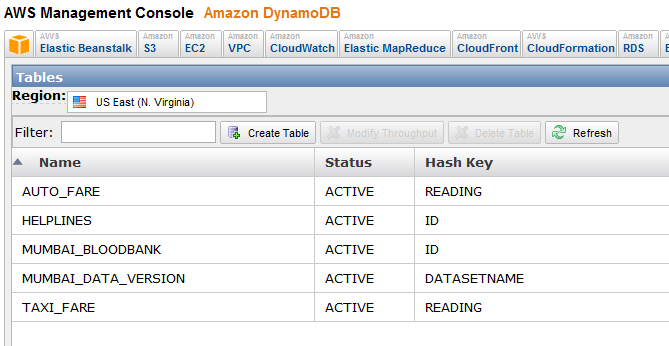
**Technical Implementation**

The solution has been implemented by making use of two exciting technologies or should I say **three**? ☺

* Amazon DynamoDB
* NodeJS
* CloudFoundry PaaS

1. **AmazonDynamoDB**

The Amazon DynamoDB has been setup for my account with the following Tables as shown below. The screenshot below is a live snapshot of my Amazon DynamoDB console with the tables listed below. Each of these tables has been populated with the data and that is what is currently being returned by the services.



The project is provided in **Amazondynamo-Java-Project.zip** file.

The Amazon DynamoDB project is currently a Eclipse Java Project and it is solely responsible for populating the datasets as they change. This is a command line Java program that is used to populate the database.

To run the same in your environment, you will need to provide your AWS Key and Secret and make sure that the appropriate tables are setup in your Amazon DynamoDB console.

1. **Implementation of** [**http://mumbaidata.cloudfoundry.com**](http://mumbaidata.cloudfoundry.com)

This is a NodeJS application that is responsible for exposing the REST endpoints as given earlier in the document. The nice part about this is that the NodeJS App talks to the Amazon DynamoDB service.

The project is provided in **Amazondynamo-Node-Project.zip** file.

* The main file is app.js and you will find that it implements various end points like /versions, /helplines and so on.
* Each invocation of the REST endpoint results in a Amazon DynamoDB scan call that returns all the data.
* For e.g. /helplines will result in a Scan of the DynamoDB table named HELPLINES
* The JSON dataset natively returned by DynamoDB is passed on to the client

1. **CloudFoundry PaaS**

The NodeJS application is hosted on the CloudFoundry PaaS. To deploy the solution you to need to have a CloudFoundry account. Follow the typical steps via command line **vmc** to install your app from the folder in which you extract **Amazondynamo-Node-Project.zip**

**Future Plans**

The solution I have provided has been deliberately designed in the form of an API. Reasons are manifold, chief among them being that:

1. An API will result in much larger consumption in ways I might not imagine.
2. Mumbai DataSets are difficult to get. This is just a start.
3. DynamoDB offers a great way to store/update your data independent of the client consumption process.

I hope to keep adding more datasets in the future if the project takes off.