# W205 Group Project : Fall 2016

Karin, Dan, Shankar and Chandler



## Meaningful Use Case / Why Meetup?

The goal of our project is to develop an app that can measure public sentiment and interests by examining the topics of upcoming Meetup events across the United States. Especially those with multiple RSVPs.

Our choice of Meetup as a data source is due to the diversity of locales that Meetup events occur in, offering a sample of data that can target less populous areas rather than just major cities. Data from Meetup will allow us to identify frequent event themes across many locations, and anticipate up and coming trends by looking at Meetups that are scheduled in the advance.

The meetup data stream allows us to meet the project requirements of:

1. Live data stream (regularly updated data)
2. Data Velocity
3. Design for data variety
4. Complex computation
5. Real time Analytics
6. Capability for Multiple data sources

Meetup Users create profiles and then join or create events. Interested members sign up for the events and meet in public to enjoy the activity together. A wide range of categories- technology, fitness, food and drink, language and culture, book clubs, career and business.   
Quick search found events for indoor soccer meetups, social entrepreneurs, bike rides, paranormal activity.  
  
Revenue from charging organizers of groups. 9.99 per mo for groups with up to 4 organizers and 50 members. As of 8/2015 had 22.77 million members in 180 country and 210,240 groups.

## Clear Business oriented purpose for the Project

Meetup topics reflect public inclination and sentiment

Measure of frequency of similar meetings across the nation will augment and outperform national polls. When meetups from small communities are analyzed, they provide the existing sentiment at the grass roots level. Unlike national polls that concentrate on the urban population, this project has the capability of being applied to micro commutites and hence can cumulatively provide exact public sentiment

## Originally Proposed Architecture



### Summary of Architectural/Functional Flow

1. Clear processing and service layers
2. Ingest Layer is well defined
3. Meetup generates data stream, data is stored in Mongo DB which mimics a data lake. Data is processed
4. All responses to Read Calls get stored in HDFS as complete JSON files in an EBS on AWS
5. Periodically, new records are pulled out and moved to a NoSQL data base on a different EBS where the JSON files are opened and a schema on write is applied to each file
6. Next, Data is taken and updated
7. Tables are queried and views are rendered

## Iterative redesign and refinement through Implementation

The design changed form as we learned much during the implementation phase. The final form lokks as in the following diagram.



The meetup stream is preprocessed at the Ingest layer. JSON stream is written to a file and individual records are then read back from the file before being put into Mongo DD. The current implementation allows for Data Cleanup at this read step and improperly formed JSON is dropped.

## Tradeoffs and Rationale of final implementaion



## Scaling the Solution





## Running the code

An AMI has been built specifically to run this project named MIDS\_Fall\_W205-4\_Meetup, start an EC2 instance using this image and open port 10000 on it if you wish to be able to connect Tableau to the Hive serving layer.

### ## Running using the startup script

After launching the instance and using ssh to connect to it as root, run the following commmands to run the end to end code:

```

git clone git@github.com:cmccann11/W205\_Group\_Project.gitcd W205\_Group\_Project

./startup.sh

```

### ## Running the pieces of the code individually

This code is intended to be run as the w205 user on the provided EC2 instance. If you've already run the startup script, then the code will be present in the w205 home directory. If not it should be copied/cloned to be made available to the w205 user.

#### ### Meetup data collection

The code for the Meetup data collection is located in `src/Meetup/`. This directory contains code to invoke and run the Meetup open events API. This API searches for recent and upcoming public events hosted by Meetup groups. Its search window is the past one month through the next three months. Open Events searches for current events by location, category, topic, or text, and only lists Meetups that have 3 or more RSVPs.BEFORE STARTING: These scripts use a common mongo\_config.py file containing mongo configuration.

To ensure that this config file is available to be included, the following command must be run:

```

export PYTHONPATH=<path/to/repository>/src

```

Note this command only needs to be run once per session.

#### #### Using the python interpreter

```

$ export PYTHONPATH=<path/to/repository>/src

$ cd src/Meetup

$ python3

>>> import mu\_spout\_api

>>> from mu\_spout\_api import mu\_spout\_on

>>> import mu\_mdb\_client

>>> mu\_spout\_on()

```

This call will generate a open events stream from 6 chosen cities with the call running

in an endless loop. The call between APIs is time delayed by 20 seconds so that we do not

exceeed the 200 API calls per hour.

#### #### To run the program from console

1). Option 1. Call the mu\_spout\_on() function by:

```

export PYTHONPATH=<path/to/repository>/src

python3 mu\_start.py

```

2). Run the mu\_spout\_on funtion directly by ``` export PYTHONPATH=<path/to/repository>/src

python3 mu\_spout\_api.py

```

#### To clean up MongoDB after any testing, use the mu\_mdb\_client.py

``` export PYTHONPATH=<path/to/repository>/src python3 mu\_mdb\_client.py ```THIS CALL WILL DELETE ALL STORED ENTRIES IN THE MONGODB. USE WITH CAUTION.

### Meetup data transformation

The meetup data transformation and creating of the serving layer is handled using spark. The code located in `src/spark`. The code will read that data in from the Mongo data store, clean and store the data in Hive where it can be served up to other applications

## APPENDIX :

