Synopsis

This document is created to help us with confusions of data integration regarding the front-end Angular Stack, back-end server API, and back-end cloud storage block (for storing video files).

Integration Plan

Front-End: Angular.js

Server Communications: Node.js & Express.js

Data Store for Client Information + Other small info: MongoDB Server

Data Store for Video Snapshots: Initially Local Hard Drive, then converted to online cloud storage.

Phase 1 – We will need a computer that is running 2 **localhost** servers all with different ports to test our app works locally. The computer will also need storage space to store video snapshots.

* Deploy the **Central** Angular App on a **localhost** with a port.
* Deploy the **Central** Node.js Server with Express on **localhost** with a port. It is configured with API listeners to communicate with the **Central** Angular App.
* Deploy a **client** computer who is running their webcam and sending their webcam stream via **API Call** to **localhost** Node.js Server.
* The **Central** Angular App will continuously do API fetch call on **Central** Node.js server to retrieve live stream data for **various** client webcam streams
* The **Central** Angular App will then display the webcam stream videos onto the Page
* The **Central** Node.js Server will automatically run motion detection algorithm on the webcam stream. Video snapshot will be stored on **Local Hard Drive** for now.
* The **Central** Angular App will perform API call to retrieve video snapshots if the user navigates to video snapshots page.

Phase 2 – The Central Angular App & Node.js App will be deployed via **Digital Ocean OR Heroku**. **ANY** amount of client computers with webcams hooked on can send their live stream via API call to Node.js App. Digital Ocean Cloud Storage will store video snapshots. A MongoDB Database will also be deployed on Digital Ocean.

* Deploy the **Central** Angular App on **Digital Ocean OR Heroku** (I have experienced using Heroku before)
* Deploy the **Central** Node.js Server with Express on **Digital Ocean OR Heroku** with their specified port. It is configured with API listeners to communicate with the **Central** Angular App.
* Deploy a **client** computer who is running their webcam, and sending their webcam stream via **API Call** to **Digital Ocean OR Heroku** Node.js Server
* The **Central** Angular App will continuously do API fetch call on **Central** Node.js server to retrieve live stream data for **various** client webcam streams
* The **Central** Angular App will then display the webcam stream videos onto the Page
* The **Central** Node.js Server will automatically run motion detection algorithm on the webcam stream. Video snapshot will be stored on **Digital Ocean Cloud Storage**
* The **Central** Angular App will perform API call to retrieve video snapshots if the central user navigates to video snapshots page.