**PROJECT 2 PROPOSAL**

Project Name: **Global Satellite Launch from 1970 – 2020**

Team Members: **Ekin Kaplan, O.J. Ndebbio, Henry Le**

**SECTION A - WORKFLOW PROPOSAL:**

**SECTION B - DATA EXPLORATION:**

1. **REQUIRED**
2. Overview of 40+ years of satellite launches? (E. Kaplan)
   1. X :: years / decades, months, days
   2. Y :: counts
   3. What need to be done:
      * 1. write the API to extract just the Launch\_Date column
        2. This would return just the list of all date
        3. For example: [01/04/1975, 04/24/1981, ...]
        4. Use JS to extract day, month, year => append to three object with key=day/month/year and value is the value counts (need for loop and append to three objects)
        5. Do visualization of multiple x-axes: date, month, year
        6. Y-axis: single, representing the counts (refer to 16-D3/3/Activities/12 or Henry’s GitHub for D3-project
3. Plot satellite launch by countries: who has the most satellites (owner column) (O.J.)
   1. X :: country names – radial stacked bar chart (switch to simple bar chart if spending too much time and codes don’t work)
   2. Y :: counts
   3. What to do: similar to #1, step i -> iv with key is the country names, and value is the count
4. Leaflet: (O.J.)
   1. Comparing 1970 – 2020: slider over the map
   2. Choropleth maps (in total): use the data from #2
   3. Getting the country names and value count from # 2
5. Build demographic box for each satellite name: (H. Le)
   1. Satellite\_Names, Country\_of\_Operator\_Owner, Launch\_Date, Purpose, Period, Orbit Classes, Mass, Lauch\_Site
   2. Create a Python Table to contain just the above parameters
   3. How to do it:
      * 1. Use Python/ SQL Alchemy to extract the columns, then export to csv, then load manually into SQLite table (Creating the class in Python and auto loading DataFrame into SQLite may have some issue with DataType mismatch – not going to that route)
        2. Write the API for connecting to this table
        3. JS to load the demographic table based on satellite name
6. **OPTIONAL**
7. Plot satellite counts vs. purpose (technology, communication), etc.:
   1. X :: purpose of usage
   2. Y :: the counts
8. Plot satellite counts vs. users (civil, commercial, military), etc.:
   1. X :: purpose of usage
   2. Y :: the counts
9. Contractor vs. count:
   1. X :: contractor
   2. Y :: counts
10. Launch site vs. count:
    1. X :: launch site
    2. Y :: counts
11. Data query table with multiple filters. Filters:
    1. Country
    2. Users
    3. Purpose
    4. Date Launch
    5. Contractor
12. Terminology for users about Aerospace

**SECTION C – TEAM TASKS:**

1. **Data Clean-up** (E. Kaplan, by 12:00PM Jun.3, 2020)
2. Rename the columns: make it more concise, replace “space” with \_
3. Replace missing data with “Zero” (number) or “Not Available” (string)
4. Delete all “Source” columns
5. **Writing the Flask Cap, creating routes for data retrieval from SQLite**
   1. Rendering the HTML (O.J., by Jun. 5, 2020)
   2. Create the API Route for data (H. Le, by Jun. 5-6, 2020)
      1. Load SQLite DB
      2. Convert it to JSON format
6. **Build baseline HTML/ CSS** (E. Kaplan, O.J.)
7. **Build JavaScript** (All)
   1. Build drop down box with Satellite names (E. Kaplan)
   2. Filter data based on Satellite Names (H. Le)
   3. Build Demographic Info (O.J.?)
   4. Plots all data exploration (All)
8. **Readme.MD** (E. Kaplan, H. Le)

**SECTION D – LEFT OVER TO DO: (updated :: Jun, 10, 2020)**

* 1. Project Description (As team) (1hr, target to be done by June. 14)
  2. Team page (Ekin) => use bootstrap (2 days, Jun 12)
  3. Contact Page (Henry) : more like a drop down box (1 day, by Jun. 11)
  4. DataTable webpage (OJ) => make the JS works, so that it can filter the data (2 days, Jun 12)
  5. Banner more interactive => may be using carousel or appear text with D3 and JS timer (Henry Le) (vanilla JS) (by Jun. 14)
  6. Visualization page :: (3 days by Jun. 13 the lastest)
  7. API page (with Nav Bar)
  8. Compile => push to AWS (1 day, by Jun. 14)
  9. Write the readme.md (as team)

=> what to do: copy the entire index.html => delete the middle part