## Requirements from Deepak:

* Review the “Challenges and remediation” section and make changes if necessary
* Prepare a detailed technical document
* Include screenshots of web scrapping where required
* Code screenshots (examples 1 or 2 max and explain)
* Flow chart of the code development
* Best Practices that we can recommend to clients (for similar projects on code development etc.)

## Resources, Tools & Technology used

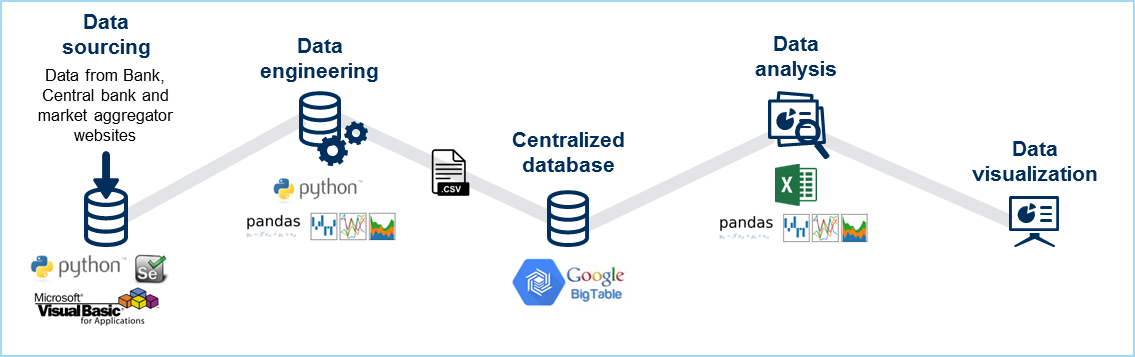
### Resources

* Used 3 analysts – 2 Data Engineer and 1 Data Expert

### Tools & Technology

* Python
  + Libraries – Pandas, Bokeh, Selenium
* C#
* PANDAS
* TABLEAU

### MAKS solution



1. Data Sourcing: Extraction of bank related data from bank and aggregator websites through automated scraping techniques using both Python and C# solutions, with the best scenario fit approach by MAKS Technical experts/Analysts. Advantages of using Python and C# are:
   1. Python has enrich library, pandas for capturing data in tables directly from the web URL
   2. Libraries like Selenium, Scrapy, Beautiful Soup are pretty straight forward and less time consuming for data scrapping
   3. Complex websites are easily doable with less development time in VBA although the run time is more.

Final solution is a tradeoff between Python and VBA, using both, as per the best scenario.

1. Data Engineering: Ensure quality data in the data lake by:
   1. Data duplicity Check at record level
   2. Introduction of Product code – A single attribute which provides information of all the important attributes to group the collected data.
   3. Data Formatting
   4. Data Filtering
   5. Data Transformation using PANDAS
2. Centralized Database
3. Data Analysis: Analysis of the quality data to get valuable insights by MAKS Domain experts/Analysts using:
   1. Descriptive Analysis using PANDAS
   2. Time Series Analysis
4. Data Visualization: Visual reporting of analyzed quality data using:
   1. BOKEH Library of Python
   2. TABLEAU

## Challenges and remediation

Challenges faced during the web scrapping project:

* IP blocked by website
* Dynamic URLs
* Identifying programs (Ex: CAPTCHA)
* Legal disclosures

We worked around these challenges by:

* Breaking the codes into smaller parts so that huge amount of data is not queried from the website at a time putting stress on the website servers
* Running the codes in frequent but multiple intervals instead of running them at one shot again avoiding consuming too much bandwidth from the websites
* Codes are written in a way that they mimic a human user by providing enough delay in going through the filters, selecting from dropdown menus etc.
* Legal disclosures are checked and made sure that we have requisite approvals to proceed
* Distributed web crawling and web scrapping by running the codes in parallel to get maximum data download speed without exerting much pressure on the servers

## Closure/Recommendations

- Technical Recommendations on the best tools to use for data scrapping

- Best practices that could be followed for similar web scrapping projects