1. Which methods or frameworks would you use?
   * I prefer using Scrapy in all my scraping projects
2. In what phase of the development would you test what?
   * Validate where the data exists
     + I would validate where exactly the data exists. My first step is always to go to the website and disable Javascript and try to find what I am looking for in the DOM. If I didn’t find it within the HTML elements, I would then try to find an API in the network and reverse engineer the API to extract the data. Sometimes it is not a trivial job and I would need to use a webdriver solution, but I always leave this as a final solution as it is slower.
   * First test case
     + As a first step of development and before designing the scraping flow, I would write the first test case based on the above validation to assure that the website structure did not change before running the spider. For example, the test case would assert the existence of main html elements on the web page
   * Develop tests for cleaning rules
     + After developing the final data model, we should have an idea on the data cleaning rules that we must validate before the data reach the final model. We then need to develop test cases to assert that those rules are actually applied
3. Which tests would you implement in the production application?
   * ETL testing: In our environment, I can think of 5 main schemas
     + RAW 🡪 store the raw data
     + WRK 🡪 store the rejected parsed data
     + STG 🡪 store the accepted parsed data after processing and applying the predefined cleaning rules.
       - E.g. Price column should be float
     + PROD 🡪 mapping the STG data into the final data model in the production application
   * RAW 🡪 WRK
     + Assert the existence of target element and if exists, then move the data to parsed. Otherwise, keep the raw data to debug on later.
   * WRK 🡪 STG
     + Apply the predefined cleaning rules and if successful, then move data to STG, otherwise keep the data in WRK. In this case, if the same failing record complies with the business rule on the next run, then this record should move to the STG schema. In other words, WRK schema should keep only the rejected data. We can have another table in the WRK schema to define the reason for rejection.
     + Using the cleaning rules to write tests on the STG to avoid False Positive results. Record passed but should not have done.
   * STG 🡪 PROD
     + Mapping the stg data to the final data model in the production database.
   * If I am using scrapy, I would directly use scrapy stats, otherwise, I would create my own logs which I can use for testing. We can also use those logs to monitor the application run. This can be very useful if we have many applications. We can monitor all the applications in one dashboard. Scrapy stats looks like this:
     + {'downloader/request\_bytes': 1024,

'downloader/request\_count': 303,

'downloader/request\_method\_count/GET': 303,

'downloader/response\_bytes': 1780,

'downloader/response\_count': 303,

'downloader/response\_status\_count/200': 303,

'downloader/response\_status\_count/404': 0,

'item\_scraped\_count': 303,

'log\_count/DEBUG': 303,

'log\_count/INFO': 0,

'memusage/max': 5200,

'memusage/startup': 90816,

'request\_depth\_max': 2,

'response\_received\_count': 303,

'scheduler/dequeued': 303,

'scheduler/dequeued/memory': 303,

'scheduler/enqueued': 303,

'scheduler/enqueued/memory': 303,

'start\_time': datetime.datetime(2017, 11, 16, 8, 58, 2, 1780)}