## **UberEats and FoodPanda Comparsion Data Pipeline AWS Cloud Watch** AWS Step-Function / Init GMT+8 10:30 daily. **AWS Lambda** Stage N / Stage Title Stage 0 / Dispatch UberEats Address Stage Description. Read address in JSON file in the environment, return them to next stage. lambda\_output(Type) \*UberEats only show visiter around 700 restaurants each time they visit. Process both sides' data. To collect more restaurants, I have to pick four other locations at East, Only process UberEats South, West, North 2.5 Km away from AppWorks School. Write DB data. Only process FoodPanda addresses(Array) Data Writes to Collection Read DB Stage 1 / Get UberEats Restaurants Info Parallel Nodes Count Use Selenium-Wire to visit https://www.ubereats.com/tw, then input address, click show more Restaurants Info button until there is no more restaurant, then parse HTML and XHR Contains both sides' data. responses for restaurants info. Only contains UberEats data. Write restaurants info to DB["ue\_list"]. Only contains FoodPanda data. restaurants\_count(Int) MongoDB Stage 1 / Get FoodPanda Restaurants Info Read address from JSON file in the environment, then send it with requests to https://disco.deliveryhero.io/listing/api/v1/pandora/vendors to retrieve restaurants info. Write restaurants info to DB["fp\_list"]. restaurants\_count(Int) Stage 2 / Dispatch Restaurants First, clear DB["{source}\_list\_temp"]. Then Read Restaurant Info from DB["{source}\_list"], and copy records written by the last stage to DB["{source}\_list\_temp"]\*. {source}\_list\_temp Restaurants Info Then create an index\_array, each element is consists of an offset and a limit. Let the next stage's Lamda knows where to query in temp collection. \*To avoid memory error on MongoDB server due to 14 Lambdas query with sort to a collection with (3000 \* (today - 2021/06/21)) records. indexes(Array) Stage 3 / Get Restaurants Detail Read Restaurants Info from DB["{source}\_list\_temp"], then use those records to retrieve detail from {source}\_detail Restaurants Detail https://www.ubereats.com/api/getStoreV1?localeCode=tw, https://tw.fd-api.com/api/v5/vendors/{uuid}. Write restaurants detail to DB["{source}\_detail"]. restaurant\_count(Int) Stage 4 / Log Step-Function Result stepfunction\_log | Step-Function Result Write a record with {"matched": False} to DB["stepfunction\_log"], which Stage 5 is listening. Stage 5 / Match and Query Google Place API Listening to DB["stepfunction\_log"], once there is a record with {"matched": False}, it will use that record's info to query data from DB["{source}\_detail"]. Then compare all restaurants with GPS coordinates and titles, after that, the process will combine records from both sources. After matched, process will update {"matched": True} to DB["stepfunction\_log"]. The Next step is query info from Google Place API. First, use former records to update if the restaurant was queried in the last 7 days. For those new restaurants, will query with their title, address , and GPS coordinates. Finally, the process will write data to DB["matched"]. {"matched": True} All Restau ant Detail matched **Website Server**

Query the latest matched restaurants.