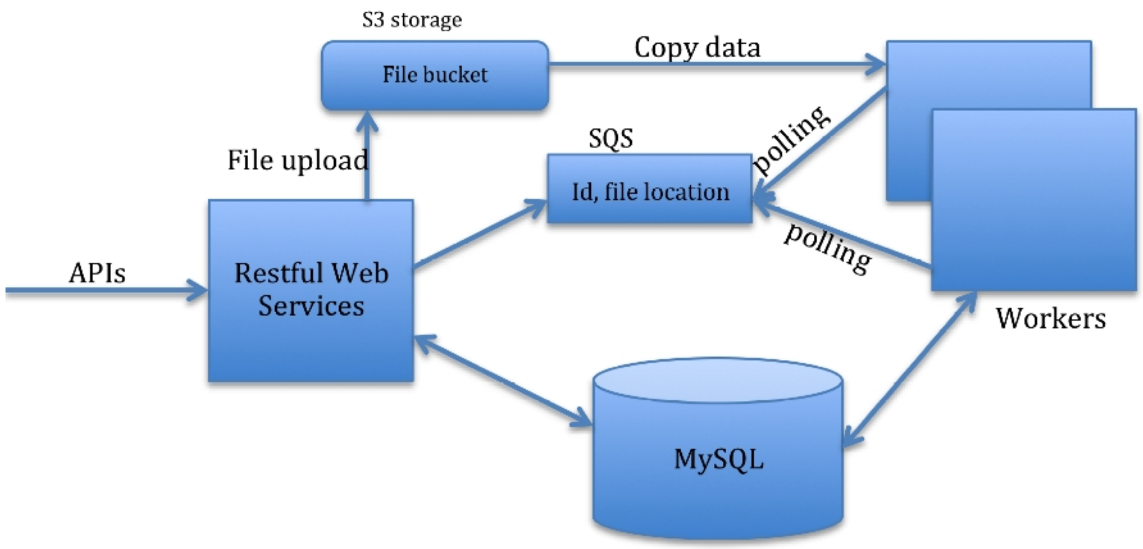
1. What is it?

Inventory management system provides a set of APIs and a web interface that aims to solve various problems faced by an Inventory/Stock manager of an SMB like updating inventory in batch, tracking the outflow(outbound) transactions, viewing and updating a particular product's quantity and adding a whole new product in the inventory.

2. How does it work ?

Below is the block diagram of the architecture of the system:



There are 2 pieces of the system, First is the online system which helps in fetching the current stock of a product and also check out the inventory when a user buys any product. I have used a RDBMS (MySQL) as all transactions in the system need to follow ACID property. I have a **inventory_detail** table which persist the state of each product, By state I mean the current inventory, the product name and its type. For auditing purposes we maintain the timestamp of updation and creation time of the product state. Apart from this I have a **inventory_transaction** table which maintains all the transaction done both inbound (when inventory is added of our system) and outbound (when inventory is checkout from our system).

The second piece is batch asynchronous system which helps to add inventory via feeds, The feed should be in csv format in which the first column is product id and the second is number of item that has to been for the particular product to our inventory. Inventory manager is supposed to upload a feed which is then uploaded to AWS S3 (Cloud based storage system). On uploading a feed we also send a event message to a cloud based distributed queue (AWS SQS). The message contains the transaction id and the details of file uploaded in S3. We have poller which continously for messages from this queue. On receiving a message, the poller reads the feed from s3. After parsing the feed from s3, it updates the product details in **inventory_detail** table. I also have additional table **inbound_transaction_status** which maintains the status of the feed processing. After a feed has been processed the poller update the state of the transaction stating either SUCCESS, FAILED or PARTIAL_SUCCESS. We have a API which helps user to check the state of the feed being processed. This API is served using **inbound_transaction_status** table.

3. How to use the application ?

Currently following five features are supported by the application in UI:

1. *Get the product detail by Id* : On entering the valid product Id the various details retaled to the product like name, category, quantity, creation time-stamp are displayed.

2. *Add a new Product* : This feature takes the details of the product like Id, name, category, quantity adds it to inventory the inventory.

3. *Update inventory* : User may want to upload a file in order to update the inventory stocks instead of adding a new product every time. This feature solves that problem. User can upload a **CSV** file containing (productId, quantity) without header and upload it which add the quantity present in the CSV file to the existing quantity corresponding to each product Id. The product Id in the CSV needs to be of type int/long. This type of transaction corresponds to the inflow/inbound transaction.
   Example for CSV format :
   ```
   11230011, 120
   12892439, 100
   13094030, 1010
   ```

4. *Checkout Inventory* : Whenever an order is placed for a particular product, its quantity needs to be deducted from the inventory if present adequately in the inventory. This feature takes product Id and quantity as input and updates the inventory (if entered quantity is less that the quantity present in the inventory. This type of transaction corresponds to outflow/outbound transaction.

5. *View transaction history of past N hours* : Use can view outflow/outbound transaction that happened in past N hours. This feature takes value for N as an input and returs all the outflow transaction details that happened in past N hours.