**Problem Statement:**

Handle high volume API calls to a platform. Provide fail-safe mechanism. Notify the platform within a day.

|  |  |
| --- | --- |
| Estimated Volume | 200 changes per minute. |
| Estimated Daily Volume | 200 \* 60 \* 24 = 288,000 |

**Assumptions:**

* Total users on the Org = 1250.

**Analysis:**

**Sync vs Async APEX:**

The requirement clearly states to notify within a day and not immediately. Also, considering the per day volume, Async Apex is the only option.

**Async APEX Governor Limits:**

|  |  |
| --- | --- |
| Async Apex | * Max allowed daily limit on this Org = 250,000. |
| Future Method | * Max allowed daily limit on this Org = 250,000. |
| Batch Apex | * Max queued or active batch jobs = 5 * Max callouts in a transaction = 100 |
| Queueable Apex | * Max active = 100 * Max callouts in a transaction = 100 |
| Flex Queue | * Max queued jobs = 100 |

**Callout Fail-Safe Mechanism:**

* The number of callouts in a single Batch or Queueable apex transaction is 100.
* Few records might fail.
* Need to provide a mechanism to differentiate the ‘pending’, ‘queued’, or ‘completed’ records.
  + Pending = Not picked for Workforce Planning API callout. Or Failed earlier callout processing.
  + Queued = Currently in one of the Apex Async jobs for processing
  + Completed = Successfully processed. Secret Key is present on the record.

**Solution:**

**Case Object:**

|  |  |  |
| --- | --- | --- |
| Closed\_By\_\_c | Lookup(User) | Stores the Closed By user. |
| WP\_Processing\_Status\_\_c | Picklist (Pending, Queued, Completed) | Indicated the Workforce Planning API callout status. |
| WP\_Secret\_Key\_\_c | Text(255) | Stores the Secret Key from Workforce Planning API. |

* Process builder on Case to update the WP\_Processing\_Status\_\_c to PENDING and set the Closed\_By\_\_c to current user (Case.LastModifiedById).

**Handling Flex Queue:**

Based on APEX Governor Limits around Callouts(max 100), batch apex (max active = 5) & flex queue (max queued = 100) and considering the sheer volume (200 per minute), there is a case to build a custom framework to handle queuing jobs in Apex Flex queue.

A mechanism to build such a framework is available at - <https://www.jitendrazaa.com/blog/salesforce/framework-to-fix-governor-limit-of-100-jobs-in-flex-queue/>

**Putting it all together:**

|  |  |
| --- | --- |
| **Apex Class** | **Function** |
| WkfcePlanScheduler implements Schedulable | * Use the AsyncApexFramework to call the batch job |
| WrkfcePlanDelegatorBatch implements Batchable | Start Method –   * Query Locator = Pick all Closed cases where WP\_Processing\_Status\_\_c = ‘Pending’ ORDER BY ClosedDate ASC * Sorting by ClosedDate ASC will pick from oldest   Execute Method –   * Update all the case records set WP\_Processing\_Status\_\_c = ‘Queued’. * Split the records in scope to chunks of 100 * Look through the list of chunks and submit a Queueable * Serialise the chunk of Cases as input to the Queueable. |
| WrkfcePlanQueueable implements Queueable, AllowsCallouts | Execute Method –   * Go over all the records in scope (Max = 100) * Perform a callout to Workforce Planning API * If response code = 200, update the WP\_Processing\_Status\_\_c to ‘Completed’ and set the WP\_Secret\_Key\_\_c. * If response code = 500 or exception occurs, update the WP\_Processing\_Status\_\_c to ‘Pending’. |

* The WkfcePlanScheduler can be scheduled as needed. Recommended = every hour.

**References & Credits:**

* https://developer.salesforce.com/docs/atlas.en-us.salesforce\_app\_limits\_cheatsheet.meta/salesforce\_app\_limits\_cheatsheet/salesforce\_app\_limits\_platform\_apexgov.htm
* https://www.jitendrazaa.com/blog/salesforce/framework-to-fix-governor-limit-of-100-jobs-in-flex-queue/
* https://salesforce.stackexchange.com/questions/236478/apex-break-list-into-multiple-small-lists-before-processing