**Config API Development guidelines**

1. APIs will be developed in different micro services based on the bounded context.
2. Vendor has to look into the business logic in the old code and write the new code.
3. Each API end point should contain only resources (nouns) no actions or verbs. The resource should always be **plural** in the API endpoint and if we want to access one instance of the resource, we can always pass the id in the URL.[HTTP methods](https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol#Request_methods) (**GET, POST, DELETE, PUT**) also called as verbs should be used to identify action to be performed.

**i.e.**

**/addEmployee**

**/getEmployee**

**/updateEmployee**

**/deleteEmployee**

**Instead of above endpoint it should be /employees .**

method **GET** path /**employees** should get the list of all employees

method **GET** path /**employees/34** should get the detail of employee having id method **DELETE** path /**employees/34** should delete company 34

method **POST** path **/employees** should add a new employee

1. When the client raises a request to the server through an API, the client should know the feedback, whether it failed, passed or the request was wrong. For this please make use of HTTP response Status code as below.

**200- OK**

**201-created**

**400- Bad request**

**401- Unauthorized**

**403- Forbidden**

**404- Not found**

**500-internal server error**

**503- Service unavailable**

1. Each API need to be versioned. Please use **URI versioning** for API.
2. Payload of each request landing on the respective microservice will be in encrypted format

**NOTE: - Encryption logic will be provided by Comviva**.

1. Once the request lands on the respective microservice it will first decrypt the encrypted payload and do the bearer code validation. If the bearer code is not supported by the system

It should throw an error with proper error code and message. **(Auth-Service)**

i.e.

**NOTE: - Error Code and message description will be provided by Comviva.**

1. After the bearer code validation Authentication and Authorization check need to be done.

If it fails at authentication or authorization system will send error response with proper error code and message. **(Auth-Service)**

**NOTE: - Error Code and message description will be provided by Comviva.**

1. Once the authentication and authorization is done, API should process the request and after successful processing it should dump the required data into the production database or fetch the required data from the production database and send success or failure response.
2. Once the required data successfully dumped into the production database there should be one KAFKA producer which will push the data into a KAFKA Topic.
3. There will be a new KAFKA Consumer which will pull the required data from the KAFKA topic and dump it into reporting DB (Postgres).
4. The Activities performed inside each API call need to be audited. Little information needs to be stored in database for audit trail reporting.

**NOTE: - Comviva will provide the format for audit trail and audit logging.**

1. Each API needs to be 100% automated. A developer has to write unit test for each of the API and the acceptance test cases provided by the tester need to be automated by him.
2. Automation done by the vendor must be compatible with the Comviva automation frame work, which is used for Mobiquity 5.0 testing.
3. API Config automation suite must replace the data builder suite, which is used for the base test environment readiness.
4. Post running the API Config automation suite on an integrated environment, all other available test suites should run as it is running on earlier platform.
5. Any fix/modification in the API Config test suite must be compatible with the existing platform and other test suites and should not hamper any of the existing functionality.
6. Once the API development is done the config api platform need to be on boarded into the Sonar, Chekmarx and CAST environment. If there is any issue found it needs to be resolved immediately. There should not be any critical and major violation exists in the system.
7. One build and deploy job need to be created in jenkin for the config api platform.

Jacooo need to be integrated into the build job to get the code coverage.

1. The new config api platform component need to be dockerized in such a way that it will be compatible with Openshift deployment.
2. All the API need to support swagger sandbox implementation.
3. All the error code and messages used in the API need to be externalized. None of them should be in properties files instead it should be put into database.
4. Logs of the config API platform should be written into the KAFKA.