REQUIREMENT ANALYSIS DOCUMENT

1.User Story : Create an api to get an account

General Request Format

```
POST /account

Accept: application/json

Authorization: Bearer w0mcJylzCn-AfvuGdqkty2-KP48=

Content-Type: application/x-www-form-urlencoded

accountId=1357902468
```

General Response Format

```
HTTP/1.1 200 OK

Content-Type: application/json; charset=utf-8

{

  "DepositAccount" : { }

}
```

Response type: DepositAccount, LoanAccount, LocAccount, or InvestmentAccount

Task:

1. Requirement Analysis – Identify the Request and Response data

2. Create Raml to get Account information

3. Create Spring boot App to get Account

a. Retrieve Data from DB table

b. Push Data to the Queue before exposing Data through API

c. Retrieve data from soap endpoint and expose as Rest API

4.Create Mulesoft App to get Account

a. Retrieve Data from DB table

b. Push Data to the Queue before exposing Data through API

c. Retrieve data from soap endpoint and expose as Rest API

5.Handle exceptions,logging and other Non Functional Requirements(Security) as requested

2.Sample Request and Response

```
Appliction Endpoint: /api/v1/account

HTTP method:  POST

Accept: application/json

Content-Type: application/json;charset = utf-8
```

Request body sample:

```
 {
       "AccountId" : "0001001DEPO"
 }
```

Response sample:

```
HTTP/1.1 200 OK

Content-Type: application/json; charset=utf-8

{
     "DepositAccount":{
             "AccountId" : "0001001DEPO",
             "AccountType" : "DEPOSIT",
             "DisplayName" : "ABC Deposit Account",
             "Status" : "OPEN",
             "Description" : "Deposit Account",
             "ParentAccountId" : "0001001",
             "Nickname" : "My Deposit Account01",
             "Currency" : "INR",
             "AccountNumber" : "0001001123",
             "InterestRate" : 3.5,
             "InterestRateType" : "FIXED",
             "MicrNumber" : "MICR0001001DEPO",
             "BalanceAsOf" : 2018-12-31T23:59:00Z,
             "CurrentBalance" : 100000.00,
            "OpeningDayBalance" : 9000,
            "AvailableBalance" : 10000,
            "AnnualPercentageYield" : 5.00,
            "InterestYtd" : 4.5,
            "MaturityDate" : 2028-12-31T21:59:59Z,
            "Term" : 10.00
          }
}
```

Response type: DepositAccount

3.Happy Flow HTTP response codes

HTTP Code	Description
200	OK ,successful response from the endpoint

4.Error Handling HTTP response codes

Error Code	HTTP Code	Conditions of occurance	Message
	400	AccountId is blank / AccountId fails the validations of max length	'Bad Request'
	401	Invalid credentials	'Unauthorized Access'
701	404	AccountId is valid but account does not exist	'Account not found'

Sample Error response :

HTTP 1.1/404 Not found

content-type: application/json

```
 {
    "Error":
          {
           "Code" : "701",
           "Message" : "Account not found"
        }
}
```

5.Entities and related Entities

-**AccountDescriptorEntity**

- -**AccountEntity**(*PK-AccountId*) --> Uses **CurrencyEntity**
  - -**DepositAccountEntity**
  - -**LoanAccountEntity**
  - -**LocAccountEntity**
  - -**InvestmentAccountEntity**

-**ErrorEntity**

6.Entity feilds and their datatypes taken in consideration

-AccountDescriptorEntity

- **-**AccountId: *Identifier (String with length 128)*
- -AccountType: *String[LOAN,DEPOSIT,INVESTMENT,LOC]*
- -DisplayName: *String*
- -Status: *AccountStatus String[CLOSED ,DELINQUENT ,NEGATIVECURRENTBALANCE ,OPEN,PAID,PENDINGCLOSE,PENDINGOPEN]*
- -Description*: String*

-AccountEntity

- -ParentAccountId: *Identifier(String with length 128)*
- -Nickname: *String*
- -Currency:***CurrencyEntity***
- -AccountNumber: *String*
- -InterestRate: *Number (decimal)*
- -Interest Rate Type: *String*
- - MicrNumber *String64*

- **CurrencyEntity**

    -CurrencyCode: *ISO4217Code*

-DepositAccountEntity

-BalanceAsOf: *Timestamp*

-CurrentBalance: *Number*

-OpeningDayBalance: *Number*

-AvailableBalance: *Number*

-AnnualPercentageYield: *Number*

-InterestYtd: *Number*

-Term: *Int*

-MaturityDate: *Timestamp*

-LoanAccountEntity

-BalanceAsOf: *Timestamp*

-PrincipalBalance: *Number*

-EscrowBalance: *Number*

-OriginalPrincipal: *Number*

-OriginatingDate: *Timestamp*

-LoanTerm*: Int*

-TotalNumberOfPayments: *Int*

-NextPaymentAmount: *Number*

-NextPaymentDate: *Timestamp*

-PaymentFrequency: *PaymentFrequency[DAILY, WEEKLY, BIWEEKLY, SEMIMONTHLY, MONTHLY,SEMIANNUALLY, ANNUALLY]*

-CompoundingPeriod: *CompoundingPeriod [DAILY, WEEKLY, BIWEEKLY, SEMIMONTHLY,*

MONTHLY,SEMIANNUALLY, ANNUALLY]

-MaturityDate: *Timestamp*

-LocAccountEntity

-BalanceAsOf: *Timestamp*

-CreditLine: *Number*

-AvailableCredit: *Number*

-NextPaymentAmount: *Number*

-NextPaymentDate: *Timestamp*

-PrincipalBalance: *Number*

-CurrentBalance: *Number*

-AvailableCash: *Number*

-InvestmentAccountEntity

-BalanceAsOf: *Timestamp*

-CurrentValue**:** *Number*

-AllowedCheckWriting**:** *Boolean*

-AllowedOptionTrade**:** *Boolean*

-AvailableCashBalance**:** *Number*

-Margin*: Boolean*

-ErrorEntity

-Code: *String*

-Message: *String*

7.Database Table references

1. **accountdescriptor**

PK -accountDescriptorId

2. **account** extends **accountdescriptor**
    PK- AccountMasterId
    FK- accDescriptorId

3. **depositaccount** extends **account**
    PK-DespositAccountId
    FK-RefAccountId

4. **investmentaccount** extends **account**
    PK-InvestmentAccountId
    FK-RefAccountId

5. **loanaccount** extends **account**
    PK-LoanAccountId
    FK-AccountId

6. **locaccount** extends **account**
    PK-LocAccountId
    FK-AccountId

7. **error**
    PK-ErrorId