Guidelines for External API Development

AB Bank Limited

Version Control

|  |  |  |  |
| --- | --- | --- | --- |
| Ver. | Date | Author | Remarks |
| 0.1 | 01-Jul-2018 | M Hedayet Hossain | Initial draft |

# Introduction

In this era of Web 2.0, open APIs, REST-based web services and fintechs, AB Bank Limited is under significant pressure to expose a set of APIs in a secured manner that will allow third-party correspondents, e.g. fintech solution providers, exchange houses, etc., to connect to the core banking service and work on associated accounts. The actions might be for status / balance enquiry of accounts, initiating payments, sending funds from overseas, etc.

In the wake of this demand, the Bank has decided to expose a set of APIs for external parties to access some key account services.

# Primary Security Considerations

Since the services will be exposed over the Internet, some measures need to be taken to minimize risk. For a secured and controlled access to sensitive information and transactional services, the following things need to be considered:

1. Encrypted connection via HTTPS
2. Non-standard port for HTTPS (not strong but an additional layer)
3. IP whitelisting
4. Auto-blocking of client IP after N number of failed attempts
5. Strong hashing (e.g. HMAC-SHA256/512) of each request and response
6. HTTP session-less communication
7. JWT (JSON Web Token) for session management

# Precautions and Security Checks

Apart from standard network-based security mechanisms, the following need to be considered. Validations need to be performed for both back-end clients and customers for maximum protection.

## Client

These considerations are to ensure that a client can be traced and controlled.

### API Client

A client means an organization who wants to access ABBL services programmatically from a back-end host. Thus the client will be pre-registered for the services with a fixed IP address and imposed limits.

1. Following will be assigned for an API client
   1. API Key – an 36-char UUID hash that acts as the login id (e.g. 63497fd4-5132-47e4-aff0-f292f04e8ad2)
   2. API Hash – acts as the login password (generated and given to client by ABBL)
   3. Shared Hash Secret – to use for generating hash for requests and responses
2. Status of the client to be maintained – active or inactive
3. IP address of the client back-end host – any request originating with the API Key from other addresses will be blocked
4. Based on the intended use, the client’s **maximum ticket size** can also be limited above which transactional requests will be denied

### JWT Security

After a successful authentication, a **token** will be return to the client for use in subsequent requests. The token will be generated conforming to the JSON Web Token (JWT – RFC 7519) open standard.

#### Token Expiry

The generated token will contain information only known to ABBL. This information will be used to identify the client in subsequent requests and also validate various parameters including **expiry** of the issued token. In case of an expired token, the client will need to authenticate again.

### Client and Merchant Accounts

A client usually has one account with the bank against which transactions would be performed with bank customers. In case the client has multiple accounts, e.g. working on behalf of other merchants, subsidiaries, then a request needs to identify the account to be operated via a **purpose code**. This will ensure that the actual account number is never present in the request.

Again, each purpose will have a maximum ticket size assigned to prevent abuse.

## Customer

These try to minimize a bank customer’s exposure to risk. Access from a client to a customer’s account will be given based on explicit permission from the customer. The following information needs to be stored for a permission:

1. Account to be accessed
2. ID of client who can access this account
3. Date, maker and checker of permission given
4. Whether **debit** request is allowed from the external client and if so then the **maximum ticket size**
5. Whether **credit** request is allowed from the external client and if so then the **maximum ticket size**
6. Whether to check requests for exact **account title**
7. Daily maximum number of transactions and total amount
8. Monthly maximum number of transactions and total amount

# REST Services

This section describes the service methods that need to be exposed to external clients.

## Security

A client should first use the login service to authenticate and obtain a JWT which will be used in all subsequent requests.

## Request Validation

All subsequent requests will contain:

* The JWT obtained from /login in Authorization header for Bearer Authentication
* A hash calculated from the request parameters and a secret key shared by both parties

A sample request header set is given below:

Content-Type:application/json

Authorization:Bearer eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJhcGlrZXllcHMiLCJleHAiOjE1MzA3MDA5NzR9.G2BVta\_bEPOs99AXP5LZFENt\_GHCHlj9pSZSU0ujmAA

## Services

### Login (/login)

|  |  |  |
| --- | --- | --- |
| Service | Request / Response | Description |
| /login | Request | apiKey - identifies the calling client API  hash - a pre-agreed secret key (UUID password) – this is separate from the req/res hash derived from parameters and secret shared hash |
|  | Response success | HTTP 200 OK  Header: Bearer JWT  Note: the obtained JWT will internally contain an expiration time after which a fresh login should be requested. |
|  | Response failure | HTTP 401 Unauthorized |

### Ping CBS (/services/ping)

Ping service for checking availability of system.

#### Request:

hash (string): HASH based on agreed secret ,

message (string): Message to get back in response

#### Response:

message (string): The exact message sent to me ,

requestIp (string): Your IP ,

serverTime (string): Server time ,

cbsConnection (string): CBS connection status

### Verify Account (/accounts/verify)

Verify if an account exists, allowed for API operations and is in an operational status.

#### Request

hash (string): HASH based on agreed secret ,

accNo (string): Bank account number ,

accName (string): Account title to be matched (max. 16 characters)

#### Response (HTTP Status)

200 OK (account ok for transaction)

451 Unavailable For Legal Reasons (e.g. account blocked, deceased, etc.)

404 Not Found (account not found)

500 Internal Server Error (unknown error)

### Account Balance (/accounts/balance)

Get balance information of an account.

#### Request

hash (string): HASH based on agreed secret ,

accNo (string): Bank account number ,

accName (string): Account title to be matched (max. 16 characters)

#### Response

status (string): Account status ,

ledgerBalance (string): Ledger balance,

statusBalance (string): Status balance ,

availableBalance (string): Available balance

### Transaction (/transactions/credit or /transactions/debit)

Transfer specific amount within limits from/to merchant account to/from customer account. Accounts must be pre-registered for debit transactions. For credit, accounts may be pre-registered or any account may be allowed depending on the client.

#### Request

hash (string): HASH based on agreed secret ,

txRef (string): An unique reference of length 15 for tracking the transaction ,

accNo (string): Bank account number ,

accName (string): Account title to be matched (max. 16 characters) ,

amount (integer): Transaction amount. NOTE: amounts must be in integer format without any decimals, e.g. 1.50 => 150 ,

narrative (string): User narrative / remarks (max. 105 characters)

#### Response

200 OK (transaction completed successfully)

400 Bad Request (invalid hash)

404 Not Found (account not found)

401 Unauthorized (account not allowed for the client / merchant)

409 Conflict (transaction with given reference code already exists)

500 Internal Server Error

### Verify Transaction (/transactions/verifyDebit or /transactions/verifyCredit)

Verify if a debit issued previously is completed or not.

#### Request

hash (string): HASH based on agreed secret ,

txRef (string): An unique reference of length 15 for tracking the transaction ,

accNo (string): Bank account number

#### Response

200 OK (transaction completed successfully)

400 Bad Request (invalid hash)

404 Not Found (account or transaction not found)

500 Internal Server Error

### Transaction Rollback (/transactions/rollbackCredit or /transactions/rollbackDebit)

Rollback a previously issued debit or credit transaction. The transaction must exist. A reversal transaction will be generated.

Request and Response are same as debit / credit – only the accounts will be switched internally.