PowerFactor Documentation

Release latest

Dec 01, 2020

Implemantation

CHAPTER 1

GettingStarted

Enterprises need secure ways to authenticate customer identity online through their digital channels, this is a feature which highly differentiates companies over their digital processes.

CHAPTER 2

Implemantation

This section aims to be a guide for users how to implement PowerFactor. These implementations refer to the tasks that must be completed to successfully enable PowerFactor in a cloud computing environment.

2.1 Core API Implementation

2.1.1 GetActivationOtp

This method is used to generate otp during new user activation. Generated otp should validate via client sdk.

`GenerateActivationOtpResponse` GetActivationOtp(`GenerateActivationOtpRequest` request)

Parameters in Request	
CustomerId	This field is the unique id of the customer to activate.
UserCode	This field is the unique id of the customer to activate.

Response	
Otp	6 numeric character, activation otp.
Results	This field used for show error details when the an error occurred during the process.
Success	This field gives information about whether an error occurred during the process.

Sample Request 1

```
{
    "CustomerId":"8053944"
}
```

- This request use for get activation otp for customer number 8053944

Sample Request 2

```
{
    "CustomerId":"999",
    "UserCode":"oyenigun"
}
```

- This request use for get activation otp for customer number 8053944 and usercode “oyenigun”.

Sample Success Response

```
{
    "Otp":"958495",
    "Results":[],
    "Success":true
}
```

Sample Failure Response

```
{
    "Otp":null,
    "Results":[
        {
            "ErrorCode":"28",
            "ErrorMessage":"Invalid request.",
            "ErrorMessageDetails":null,
            "Exception":null,
            "Params":null,
            "Severity":3,
            "IsFriendly":true
        }
    ],
    "Success":false
}
```

2.1.2 VerifyLoginOtp

This method is used to verify to otp which received from the client sdk during login process.

`VerifyLoginOtpResponse` VerifyLoginOtp(`VerifyLoginOtpRequest` request)

Parameters in Request	
LoginOtp	This otp contains user and session data. After get this value fromclient sdk, should be validate via this method while login process.

Response	
Value	Boolean value, otp is valid or not.
Results	This field used for show error details when the an error occurred during the process.
Success	This field gives information about whether an error occurred during the process.

Sample Request 1

```
{
    "LoginOtp":"a4bab53f-f313-42b1-9502-d1da3c0e8cfc"
}
```

- This request use for validate to requested login.

Sample Success Response

```
{
    "Value":true,
    "Results":[],
    "Success":true
}
```

Sample Failure Response

```
{
    "Value":false,
    "Results":[
        {
            "ErrorCode":"29",
            "ErrorMessage":"Invalid login otp.",
            "ErrorMessageDetails":null,
            "Exception":null,
            "Params":null,
            "Severity":3,
            "IsFriendly":true
        }
    ],
    "Success":false
}
```

2.1.3 StartTransaction

This method is used to start transaction approval.

`StartTransactionResponse` StartTransaction(`StartTransactionRequest` request)

Parameters in Request	

TransactionContent	**This field should contain the data to be approved on the transaction confirmation screen. This data can**
	`{`
	`"CustomerId":8053944,`
	`"UserCode":"oyenigun",`
	`"Transaction":"Web Login",`
	`"Date":"2017-04-24"`
	`}`

TransactionName	**This String field used to distinguish the transaction. Example:** `"MoneyTransferToIBAN"`
TransactionType	This field should be "TransactionApprovement(1)" for transactions which starting from the mobile device and only being approved on that device. It should be "MobileApprovement(2)" for transactions which starting from the any channel and being approved on customers all device.

TimeoutDuration	**Set different timeout duration as "seconds" for all transactions is available. Example:**
	`50 = 50 seconds`
	`180 = 3 minutes`

CancelPendingTransactions	This bool field should be "true" for the cancel all waiting transactions which started for same customer and same channel while new transaction start. This field should be "false" for multiple transaction approvement at the same time.
TransactionOwner	**OwnerKey** : This field should use for transactions which starting from the mobile device and only being approved on that device. This value can get from "getOwnerKey()" method on mobile Sdk's. **CustomerId, UserCode** : This fields use for "Mobile Approval" transactions. (Web, ATM etc.based) All devices of this customer can approve transaction.

Response	
TranscationToken	**This value is unique and created after transaction started. Example:** `"ABHfeMYffrJ2HhXg5RvlCw=="`
Results	This field used for show error details when the an error occurred during the process.
Success	This field gives information about whether an error occurred during the process.

Sample Request 1

```
{
    "TransactionOwner":{
        "OwnerKey":"20170419-e0240167-ddd6-4b5d-953d-f2dab9d00340"
    },
    "TransactionContent":"ENCRYPTEDCONFIRMATIONDATA",
    "TransactionName":"MOBILEHAVELE",
    "TransactionType":1,
    "TimeoutDuration":60,
    "CancelPendingTransactions":true
}
```

- This request start “MOBILEHAVELE” transaction. And customer can approve this transaction from only this device.
- This transactions timeout duration is 60 seconds. All waiting transactions which started before from this device to be cancelled while this transaction starting.

Sample Request 2

```
{
    "TransactionOwner":{
        "CustomerId":999,
        "UserCode":"oyenigun"
    },
    "TransactionContent":"ENCRYPTEDCONFIRMATIONDATA",
    "TransactionName":"WEBLOGIN",
    "TransactionType":2,
    "TimeoutDuration":180,
    "CancelPendingTransactions":true
}
```

- This request start “WEBLOGIN” transaction. And customer can approve this transaction from his all devices which activated before.

Sample Success Response

```
{
    "TranscationToken":"yGRDXgv4KRGRqJHqK5bLUQ==",
    "Results":[],
    "Success":true
}
```

Sample Failure Response

```
{
    "TranscationToken":null,
    "Results":[
        {
            "ErrorCode":"22",
            "ErrorMessage":"Invalid session. ",
            "ErrorMessageDetails":"Active session not found. ",
            "Exception":null,
            "Params":null,
            "Severity":3,
            "IsFriendly":true
        }
    ],
    "Success":false
}
```

2.1.4 CheckTransactionApproval

This method is used to check started transaction’s current status. In case of the approval timeout, the status changes automatically to “Expired”.

`CheckTransactionApprovalResponse` CheckTransactionApproval(`CheckTransactionApprovalRequest` request)

Parameters in Request	
TransactionToken	The token of the transaction to be check status.

Response	

Status	**This value is the current status of transaction.**
	`Rejected = 0,`
	`Approved = 1,`
	`WaitingApproval = 2,`
	`ErrorOccurred = 3,`
	`Expired = 4,`
	`Fraud = 5,`
	`Cancelled = 6`
	"WaitingApproval" is the initial status of the transaction.

Results	This field used for show error details when the an error occurred during the process.
Success	This field used for show error details when the an error occurred during the process.

Sample Request

```
{
    "TransactionToken":"3D+OwVd3nOH3KNHCK3H12Q=="
}
```

Sample Success Response

```
{
    "Status":4,
    "Results":[ ],
    "Success":true
}
```

Sample Failure Response

```
{
    "Status":0,
    "Results":[
        {
            "ErrorCode":"510",
            "ErrorMessage":"Transcation not found.",
            "ErrorMessageDetails":null,
            "Exception":null,
            "Params":null,
            "Severity":1,
            "IsFriendly":true
        }
    ],
    "Success":false
}
```

2.1.5 CancelTransaction

This method is used for cancel to pending transaction approvals.

`CancelTransactionResponse` CancelTransaction(`CancelTransactionRequest` request)

Parameters in Request	
TransactionToken	The token of the transaction to be canceled.

Response	

Status	**This value is the current status of transaction.**
	`Rejected = 0,`
	`Approved = 1,`
	`WaitingApproval = 2,`
	`ErrorOccurred = 3,`
	`Expired = 4,`
	`Fraud = 5,`
	`Cancelled = 6`

Results	This field used for show error details when the an error occurred during the process.
Success	This field gives information about whether an error occurred during the process.

Sample Request

```
{
    "TransactionToken":"3D+OwVd3nOH3KNHCK3Hl2Q=="
}
```

Sample Success Response

```
{
    "Status":6,
    "Results":[],
    "Success":true
}
```

Sample Failure Response

```
{
    "Status":0,
    "Results":[
        {
            "ErrorCode":"510",
            "ErrorMessage":"Transcation not found.",
            "ErrorMessageDetails":null,
            "Exception":null,
            "Params":null,
            "Severity":1,
            "IsFriendly":false
        }
    ],
    "Success":false
}
```

2.2 Android SDK Implementation

2.2.1 Init

This method is used for sdk initialization.

`void` Init(Context context)

Method Parameters	
Context	Active context. If active context changes, Init() should be called again.

2.2.2 ActivateDevice

This method is used for activate device on powerfactor.

`void` ActivateDevice(String otp, PWFUserContract userData, String pin, PWFCallBack<T> callback)

Method Parameters	
otp	This parameter is the activation otp sent to the customer via SMS.
userData	Customer unique data as **PWFUserContract**.
pin	The 6 digit value that the user input.
callback	This callback parameter is called automatically when the operation completed. **PWFCallBack<T>** is the interface for error & callback management.

2.2.3 Login

This method used for secure login via PowerFactor with just only 6 digits pin.

`void` Login(PWFUserContract userData, String pin, PWFCallBack<PWFLoginResponse> callback)

Method Parameters	
userData	Customer unique data as **PWFUserContract**.
pin	The 6 digit value that the user input.
callback	This callback parameter is called automatically when the Login completed. PWFCallBack<T> is the interface for error & callback management.

2.2.4 Logout

This method used for secure logout.

`void` Logout(PWFCallBack<PWFLoginResponse> callback)

Method Parameters	
callback	This callback parameter is called automatically when the Login completed. PWFCallBack<T> is the interface for error & callback management.

2.2.5 GetUserList

This method used for get activation list on current device.

ArrayList<PWFUserContract> GetUserList()

Method Result	
ArrayList<PWFUserContract>	List of **PWFUserContract**.

2.2.6 ChangePin

This method used to change the user's pin.

`void` ChangePin(PWFUserContract userData, String oldPin, String newPin, final PWFCallBack<PWFResponseModelBase> callback)

Method Parameters	
userData	Customer unique data as **PWFUserContract**.
oldPin	The 6 digit current pin value.
newPin	The 6 digit new pin value.
callback	This callback parameter is called automatically when the operation completed. **PWFCallBack<T>** is the interface for error & callback management.

2.2.7 DeleteUser

This method used for delete activation from current device.

`void` DeleteUser(PWFUserContract userData, final PWFCallBack<PWFResponseModelBase> callback)

Method Parameters	
userData	Customer unique data as **PWFUserContract**.
callback	This callback parameter is called automatically when the operation completed. **PWFCallBack<T>** is the interface for error & callback management.

2.2.8 Common Types

PWFUserContract

Properties	
CustomerId	This field should be unique identifier for customer.
UserCode	This field should be unique identifier for customer. UserCode should use when there is more than one person using the same customer number.
UserFullName	User Name and Surname. (Optional)

PWFCallBack<T>

`interface` PWFCallBack<Response extends PWFResponseModelBase>

Methods	
onCompleted(T)	This method triggered automatically when the operation completed.
onFailure(PWFCallBackError)	This method triggered automatically when the operation failure with error details.

2.3 iOS SDK Implementation

2.3.1 Init

This method is used for sdk initialization.

`func` Init ()

ActivateDevice

This method used for activate device on powerfactor.

`func` activateDevice(_ otp:String, userData: PWFUserContract, pin:String, onActivationSuccess:@escaping ()-> Void, onActivationFailure:@escaping ([Result]) ->Void)

Method Parameters	
otp	This parameter is the activation otp sent to the customer via SMS.
userData	Customer unique data as **PWFUserContract**.
pin	The 6 digit value that the user input.
onActivationSuccess	This callback parameter is called automatically when the operation success.
onActivationFailure	This callback parameter is called automatically with error details ([Result]) when the operation failed.

2.3.2 Login

This method used for secure login via PowerFactor with just only 6 digits pin.

`func` login(_ userData: PWFUserContract, pin: String, onLoginSuccess:@escaping (_ loginOtp: String)-> Void, onLoginFailure:@escaping ([Result]) ->Void)

Method Parameters	
userData	Customer unique data as **PWFUserContract**.
pin	The 6 digit value that the user input.
onActivationSuccess	This callback parameter is called automatically when the operation success.
onActivationFailure	This callback parameter is called automatically with error details ([Result]) when the operation failed.

2.3.3 Logout

This method used for secure logout.

`func` logout(_ logoutSuccess: @escaping () -> Void = { })

Method Parameters	
LogoutSuccess	This callback parameter is called automatically when the logout success.

2.3.4 GetUserList

This method used for get activation list on current device.

`func` getUserList() -> [PWFUserContract]?

Method Result	
[PWFUserContract]	Array of **PWFUserContract**.

2.3.5 ChangePin

This method used to change the user's pin.

`func` changePIN (_ userData: PWFUserContract, pin: String, newPIN : String, onPINChanged:@escaping ()-> Void, onPINChangeFailure:@escaping ([Result]) ->Void)

Method Parameters	
userData	Customer unique data as **PWFUserContract**.
pin	The 6 digit current pin value.
newPin	The 6 digit new pin value.
onPINChanged	This callback parameter is called automatically when the operation success.
onPINChangeFailure	This callback parameter is called automatically with error details ([Result]) when the operation failed.

2.3.6 DeleteUser

This method used for delete activation from current device.

`func` deleteUser(_ userData: PWFUserContract) -> Bool

Method Parameters	
userData	Customer unique data as **PWFUserContract**.

2.3.7 BeginTransaction

This method is used for begin the transaction.

`func` beginTransaction(_ onTransactionBegin:@escaping (_ data: PWFTransactionContract)-> Void, onTransactionFailure:@escaping ([Result]) ->Void = { (error:[Result]) in })

Method Parameters	
onTransactionBegin	This callback parameter is called automatically when the transaction begins.
onTransactionFailure	This callback parameter is called automatically when the transaction failure.

2.3.8 CompleteTransaction

This method is used for complete transaction. Started transactions should complete even if unexpected cases.

`func` completeTransaction(_ status: TransactionStatusNames, onTransactionFailure:@escaping ([Result]) ->Void)

Method Parameters	
status	Transaction Status. (Approved, Rejected, Timeout vs. . . .)
onTransactionFailure	This callback parameter is called automatically when the transaction failure.

2.3.9 Common Types

PWFUserContract

Properties	
CustomerId	This field should be unique identifier for customer.
UserName	This field should be unique identifier for customer. UserName should use when there is more than one person using the same customer number.

Result

Method Parameters	
ErrorCode	Unique error code.
ErrorMessage	User friendly error message.
Exception	Exception details.
IsFriendly	Is Exception or user friendly error?