

HOW TO

INTEGRATE UPAYCARD AS A PAYMENT METHOD

Contents

| | |
|---|----|
| 1. DESCRIPTION | 4 |
| 2. SECURITY | 4 |
| 3. CUSTOMER DEPOSIT FLOW | 5 |
| 3.1. Customer has to choose to pay with UpayCard | 5 |
| 3.2. Customer has to enter his Account number or Username | 5 |
| 3.3. Customer has to confirm his deposit | 6 |
| 3.4. Completed transaction | 7 |
| 3.5. Programming in PHP | 8 |
| 3.6. What calls to use from API documentation | 9 |
| 3.6.1. InitializeTransfer | 9 |
| 3.6.2. FinishTransfer | 10 |
| 3.6.3. GetTransactionStatus | 12 |
| 4. NEW CUSTOMER REGISTRATION WITH REFERALL ID | 13 |
| 4.1. Merchant Referral ID | 13 |
| 4.2. User registration with merchants referral ID | 14 |
| 5. SPONSORED ACCOUNT CREATION FROM GUI | 15 |
| 6. SPONSORED ACCOUNT CREATION USING API | 17 |
| 6.1. CreateUser API Call | 17 |
| 6.1.1. CreateUser Request | 17 |
| 6.1.2. CreateUser JSON Request Sample | 18 |
| 6.1.3. CreateUser JSON Request Response | 19 |
| 6.1.4. CreateUser JSON Response Sample | 19 |
| 6.2. UpdateUserKyc API Call | 19 |
| 6.2.1. UpdateUserKyc Request | 19 |
| 6.2.2. UpdateUserKyc JSON Request Sample | 20 |
| 6.2.3. UpdateUserKyc JSON Response | 20 |
| 6.2.4. UpdateUserKyc JSON Response Sample | 20 |
| 6.3. Programming in PHP | 21 |
| 7. PURCHASE API | 23 |

| | | |
|--------|--|----|
| 7.1. | Purchase Flow..... | 23 |
| 7.2. | Purchase API calls..... | 26 |
| 7.2.1. | CreatePurchase | 26 |
| 7.2.2. | GetPurchaseStatus..... | 27 |
| 7.3. | Programming in PHP | 28 |
| 8. | PURCHASE URL..... | 31 |
| 8.1. | Purchase Flow..... | 31 |
| 8.2. | Programing in PHP..... | 33 |
| 8.3. | URL settings for IPN..... | 34 |
| 9. | APPENDIX A: SIGN GENERATION | 36 |
| 10. | APPENDIX B: ENCRYPT/DECRYPT INFORMATION | 36 |
| 11. | APPENDIX C: TRANSACTION CODES..... | 37 |
| 12. | APPENDIX D: USER KYC DOCUMENT TYPES..... | 38 |
| 13. | APPENDIX E: USER ID TYPES..... | 39 |
| 14. | APPENDIX F: PHP CODE EXAMPLE | 40 |
| 15. | APPENDIX G: Invalid requests error codes | 40 |
| 16. | APPENDIX H: Purchase statuses..... | 40 |
| 17. | APPENDIX J: IPN CATCHER PHP Code Example..... | 41 |

1. DESCRIPTION

UPayCard documentation provides access for the third-parties to standard functions and services implemented in UPayCard.

Merchants can build their own custom applications, tools, and services to support their programs or components of their programs.

2. SECURITY

The following aspects create additional security for each third-party:

- Both login and password are required for each call to API to authenticate the third-party.
- The password is unique per each third-party
- HTTPS Security is mandatory – an SSL certificate is required.
- All sensitive data are transmitted in an encrypted format under a 3-DES key.

3. CUSTOMER DEPOSIT FLOW

3.1. Customer has to choose to pay with UpayCard

On your website depositing methods Customer has to choose to pay with UPayCard (see Figure 1: deposit page on your website).

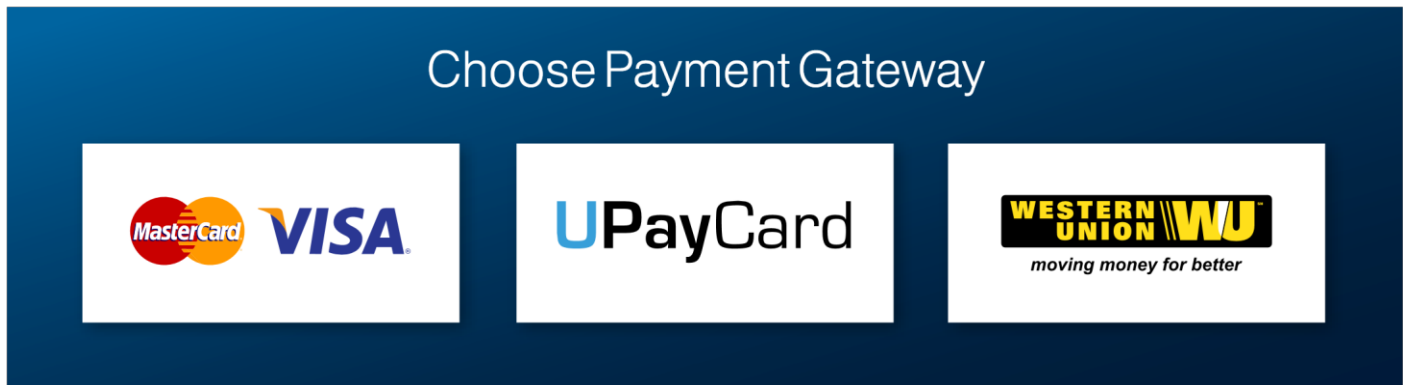


Figure 1: deposit page on your website

3.2. Customer has to enter his Account number or Username

Customer has to enter alternatively his Account number or Username (see Figure 2: UpayCard Deposit – entering username and amount).

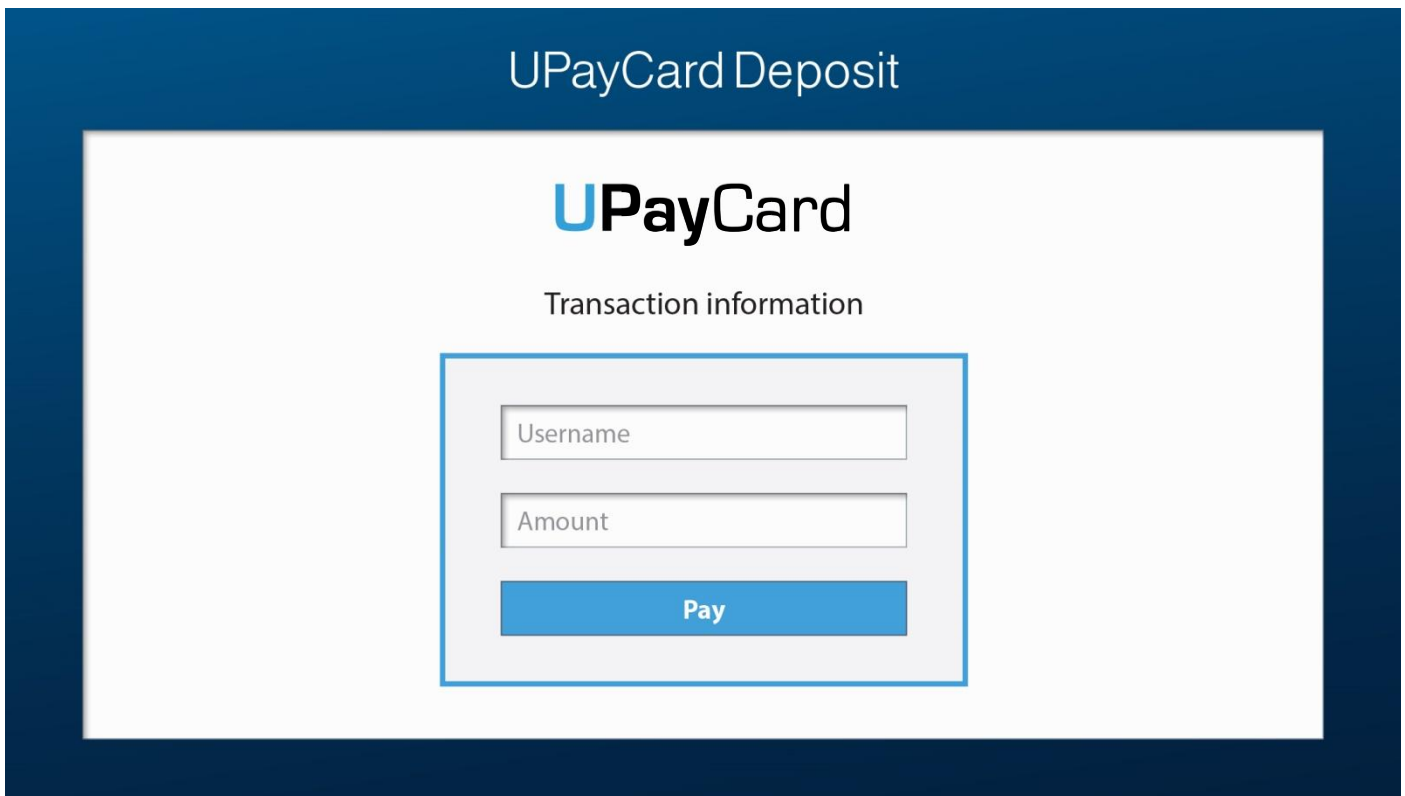
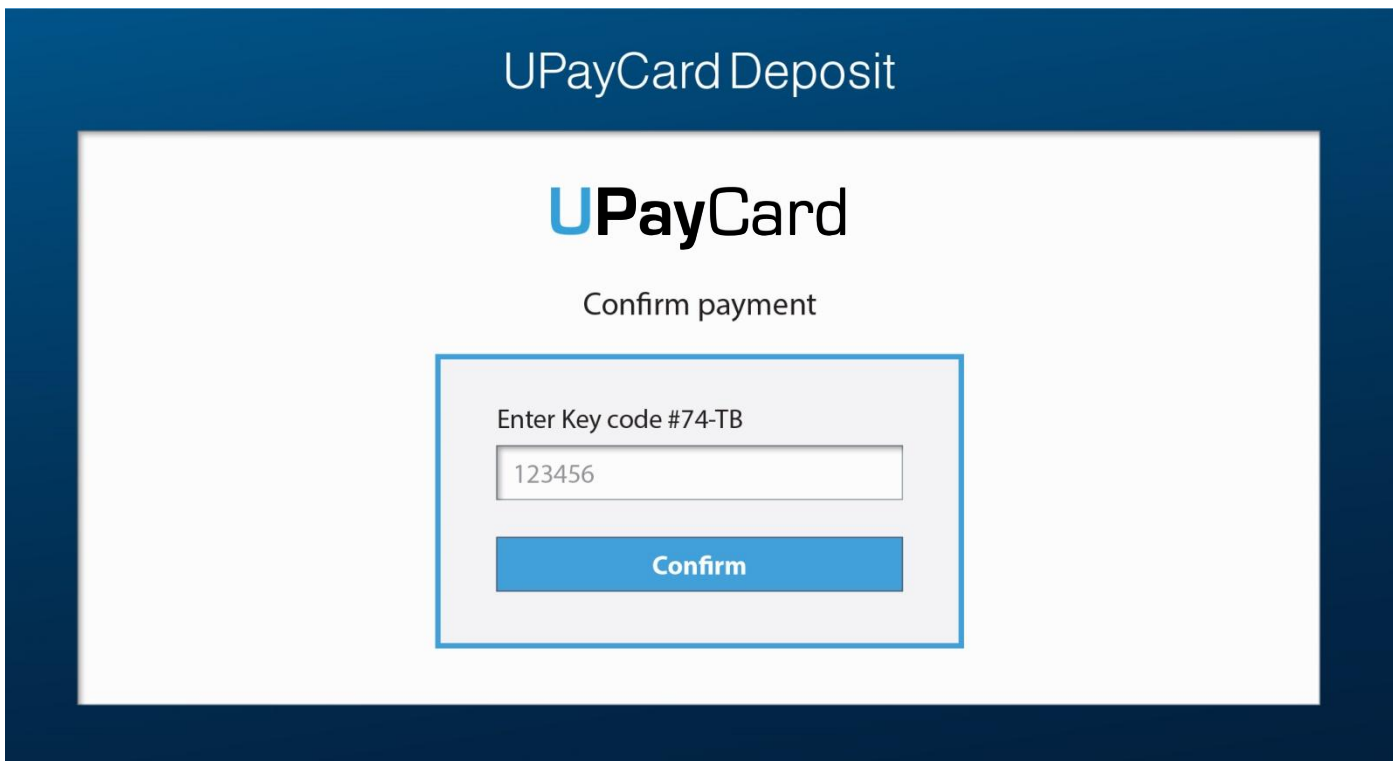
The image shows a screenshot of the UPayCard Deposit interface. At the top, the title "UPayCard Deposit" is displayed in white text on a dark blue background. Below this, the UPayCard logo is centered. Under the logo, the text "Transaction information" is shown. A light gray rectangular box contains two input fields: "Username" and "Amount". Below these fields is a blue button with the text "Pay" in white.

Figure 2: UpayCard Deposit – entering username and amount

3.3. Customer has to confirm his deposit

Customer is requested to confirm his deposit by entering one of the Keys from his KeyCode Card (see Figure 3: UpayCard deposit - confirming payment with entering Key Code).

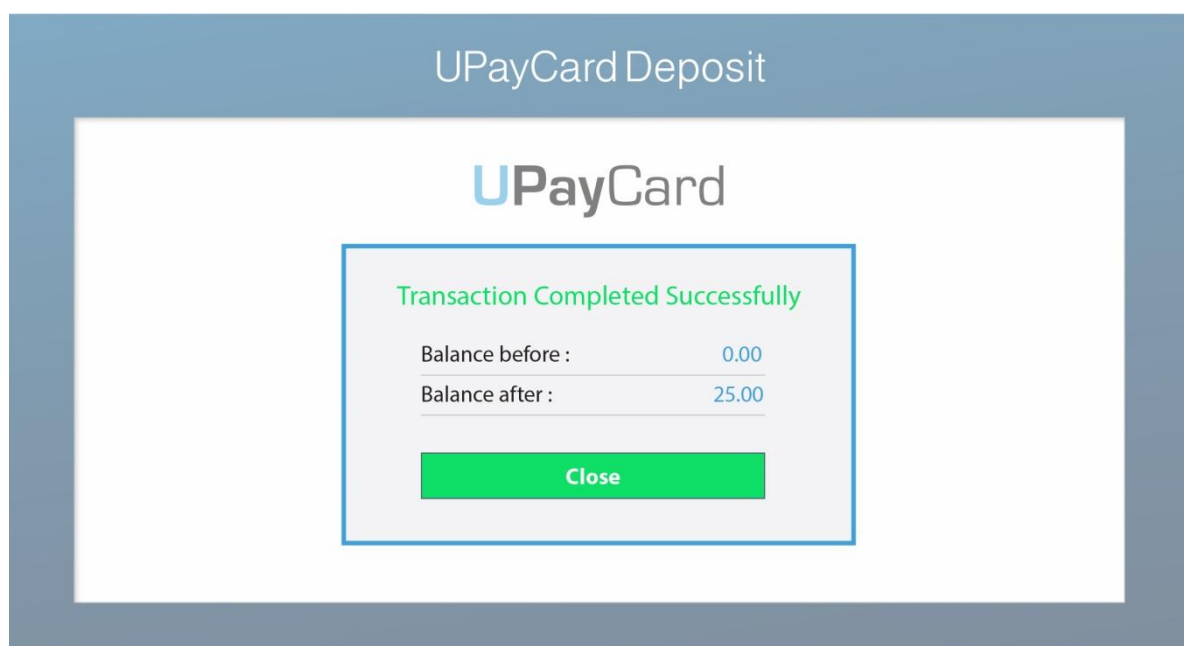


The image shows a screenshot of the UPayCard Deposit confirmation screen. The screen has a dark blue header with the text "UPayCard Deposit". Below the header, the UPayCard logo is displayed. Under the logo, the text "Confirm payment" is shown. A light gray box contains the prompt "Enter Key code #74-TB" above a text input field with the value "123456". Below the input field is a blue button labeled "Confirm".

Figure 3: UpayCard deposit - confirming payment with entering Key Code

3.4. Completed transaction

After getting and analyzing response – show successfully completed transaction.



The image shows a screenshot of the UPayCard Deposit successful transaction screen. The screen has a light blue header with the text "UPayCard Deposit". Below the header, the UPayCard logo is displayed. A light gray box contains the text "Transaction Completed Successfully" in green. Below this text, there is a table showing the balance before and after the transaction. At the bottom of the box is a green button labeled "Close".

| | |
|------------------|-------|
| Balance before : | 0.00 |
| Balance after : | 25.00 |

3.5. Programming in PHP

```
// url for sandbox https://sandboxapi.upaycard.com/api/merchant
$url = "https://api.upaycard.com/api/merchant";
$key = "your_key";
$secret = "your_secret";
$encryption_key = "your_encryption_key";
$debug_mode = true; // in live should be false
// adding library provided by UPAYCARD
require_once("Api.php");
$api = new Upaycard_Api($url, $key, $secret, $debug_mode, $encryption_key);
$receiver_account = ""; // your receiving_account number
$sender = ""; // sender account id/username/email
$amount = "100.00";
$currency = "USD"; // currency code (USD or EUR)
$order_id = ""; // order_id in your system
$description = ""; // transaction description
//
// 1 step initialize transfer and get token code
$data = $api->initializeTransfer($receiver_account, $sender, $amount, $currency,
$order_id, $description);

if ($data["status"] == "success") {
    // request your customer for $data["token_number"]
}

// 2 step finish transfer
// when user submit token code you can finish transfer
$receiver_account = ""; // your receiving_account number
$hash = $data["hash"]; // what you get from initializeTransfer() function
$token_number = $data["token_number"]; // what you get from initializeTransfer() function
$token_code = $_POST["token_code"]; // what user entered
$data_final = $api->finishTransfer($receiver_account, $hash, $token_number, $token_code);

if ($data_final["status"] == "success") {
    // now you can deposit your customer
} elseif ($data_final["status"] == "error" && $data_final["code"] == "517") {
    // error happened if error code is 517 you have to do check later
    sleep(25);
    // do sleep for few seconds or create additional functions to do check periodically
    $tx_status = $api->getTransactionStatus($data_final["transaction_id"]);

    if ($tx_status["status"] == "success") {
        if ($tx_status["transaction_status"] == "C") {
            // now you can deposit your customer
        } elseif ($tx_status["transaction_status"] == "R") {
            // tx is rejected
        } elseif ($tx_status["transaction_status"] == "P") {
```



```
        // still not processed, check with getTransactionStatus() function after some  
time  
    }  
}  
}
```

3.6. What calls to use from API documentation

3.6.1. InitializeTransfer

Prepare transfer from any user to merchant account

| | |
|-----|---|
| URL | https://api.upaycard.com/api/merchant/v/1.0/function/initialize_transfer |
|-----|---|

InitializeTransfer Request

| KEY | M | TYPE | LENGTH | DESCRIPTION |
|-------------------------|---|------|--------|--|
| receiver_account | Y | N | 11 | Your account ID for receive transfer |
| sender | Y | AN | 100 | Here can be username or user email or account number from which made transfer. If user did not have requested currency account or have several accounts then there must be account number |
| amount | Y | N | 10,2 | |
| currency | Y | A | 3 | ISO 4217 |
| order_id | Y | AN | 30 | Unique identification of request |
| description | Y | AN | 100 | Transfer description |
| key | Y | AN | 16 | Merchant API Key – provided by UPayCard |
| ts | Y | N | 10 | Request timestamp |
| sign | Y | AN | 32 | See APPENDIX A: SIGN GENERATION |

InitializeTransfer JSON Request Sample

```
{  
  "receiver_account": "1000001",  
  "sender": "user",  
  "amount": 1.01,  
  "currency": "USD",  
  "order_id": "15_20151110080801",  
  "description": "Payment for order #15",  
  "key": "_MERCHANT_KEY_",  
  "ts": "_TIMESTAMP_"
```

```
    "sign": "_SIGN_"
}
```

InitializeTransfer JSON Response

| KEY | M | TYPE | LENGTH | DESCRIPTION |
|---------------------|---|------|--------|---|
| status | Y | AN | 10 | success or error |
| code | N | N | 3 | See APPENDIX C: Transaction Codes |
| msg | Y | AN | 255 | See APPENDIX C: Transaction Codes |
| description | N | AN | | Detailed explanation of error |
| order_id | N | AN | 100 | UniqueID of request |
| hash | N | AN | | |
| token_number | N | AN | 10 | |

InitializeTransfer JSON SUCCESS Response Sample

```
{
  "status": "success",
  "msg": "Transfer initialized.",
  "order_id": "15_20151110080801",
  "hash": "a64fb511c885f9aeff211f7bfefc5648",
  "token_number": "48-QI"
}
```

InitializeTransfer JSON FAILED Response Sample

```
{
  "status": "error",
  "code": 509,
  "msg": "Define sender account",
  "description": "Did not found active account for currency",
  "hash": "5640bf0cb3e5c"
}
```

3.6.2. FinishTransfer

Finish prepared transfer from any user to merchant account. Request user to enter his Key Code for given token_number.

URL

https://api.upaycard.com/api/merchant/v1.0/function/finish_transfer

FinishTransfer Request

| KEY | M | TYPE | LENGTH | DESCRIPTION |
|------------------|---|------|--------|---|
| receiver_account | Y | N | 11 | Your account ID for receive transfer |
| hash | Y | AN | | hash code given in InitializeTransfer success response |
| token_number | Y | AN | 10 | token_number given in InitializeTransfer success response |
| token_code | Y | N | 6 | User's entered Key Code for given token_number |
| key | Y | AN | 16 | Merchant API Key – provided by UPayCard |
| ts | Y | N | 10 | Request timestamp |
| sign | Y | AN | 32 | See APPENDIX A: SIGN GENERATION |

FinishTransfer JSON Request Sample

```
{
  "receiver_account":"1000001",
  "hash":"5640bf0cb3e5c",
  "token_number":"48-QI"
  "token_code":"123456",
  "key": "_MERCHANT_KEY_",
  "ts": _TIMESTAMP_,
  "sign": "_SIGN_"
}
```

FinishTransfer JSON Response

| KEY | M | TYPE | LENGTH | DESCRIPTION |
|----------------|---|------|--------|---|
| status | Y | AN | 10 | success or error |
| code | N | N | 3 | See APPENDIX C: Transaction Codes |
| msg | Y | AN | 255 | See APPENDIX C: Transaction Codes |
| description | N | AN | | Detailed explanation of error |
| transaction_id | N | N | 11 | |
| order_id | N | AN | 100 | UniqueID of request |

FinishTransfer JSON SUCCESS Response Sample

```
{
  "status": "success",
  "code": 000,
```

```
{
  "msg": "Transaction successfully completed",
  "transaction_id": "100687",
  "order_id": "15_20151110080801",
}
```

FinishTransfer JSON FAILED Response Sample

```
{
  "status": "error",
  "code": 512,
  "msg": "Wrong Key Code provided",
}
```

If finish transaction was failed then procedure must start again from InitializeTransfer.

3.6.3. GetTransactionStatus

Get transaction status by transaction ID.

| | |
|-----|---|
| URL | https://api.upaycard.com/api/merchant/v/1.0/function/get_transaction_status |
|-----|---|

GetTransactionStatus Request

| KEY | M | TYPE | LENGTH | DESCRIPTION |
|-----------------------|---|------|--------|---|
| transaction_id | Y | AN | 11 | |
| key | Y | AN | 16 | Merchant API Key – provided by UPayCard |
| ts | Y | N | 10 | Request timestamp |
| sign | Y | AN | 32 | See APPENDIX A: SIGN GENERATION |

GetTransactionStatus JSON Request Sample

```
{
  "transaction_id": 22,
  "key": "_MERCHANT_KEY_",
  "ts": "_TIMESTAMP_",
  "sign": "_SIGN_"
}
```

GetTransactionStatus JSON Response

| KEY | M | TYPE | LENGTH | DESCRIPTION |
|-------------------------|---|------|--------|---|
| status | Y | AN | 10 | success or error |
| msg | Y | AN | 255 | See APPENDIX C: Transaction Codes |
| transaction_status | Y | A | 1 | Possible values: C: Completed P: Pending R: Rejected |
| transaction_code | Y | A | 3 | See APPENDIX C: Transaction Codes |
| transaction_status_desc | Y | N | 50 | |
| test | Y | A | 1 | Possible values: Y: Yes N: No |

GetTransactionStatus JSON Response Sample

```
{
  "status": "success",
  "msg": "Success",
  "transaction_status": "C",
  "transaction_code": "000",
  "transaction_status_desc": "Transaction successfully completed",
  "test": "N"
}
```

4. NEW CUSTOMER REGISTRATION WITH REFERRAL ID

4.1. Merchant Referral ID

In order to make new registered user as his sponsored account – merchant needs to provide registration URL with his referral ID.

Login to UpayCard as a merchant. Referral ID with referral link can be found under “My Sponsored Accounts” menu (see Figure 4: “My Sponsored Accounts” page).

URL example: https://user.upaycard.com/auth/register/r/ YOUR_REFERRAL_ID

MY SPONSORED ACCOUNTS

SEARCH ACCOUNTS

Date Created From:

Date created to:

Is Registered:

Search:

<https://user.upaycard.com/auth/register/r/0123456789>

Create New Sponsored Account

Create New Sponsored Account With Card

Create Multiple Sponsored Accounts From File

Show entries

Search:

| Actions | Username | Password | Accounts | Cards (Last used) | Cards (Envelope numbers) | Registered | Created |
|---------|----------|----------|----------|-------------------|--------------------------|------------|------------|
| | upc1 | | 1778682 | | | - | 2016-11-09 |

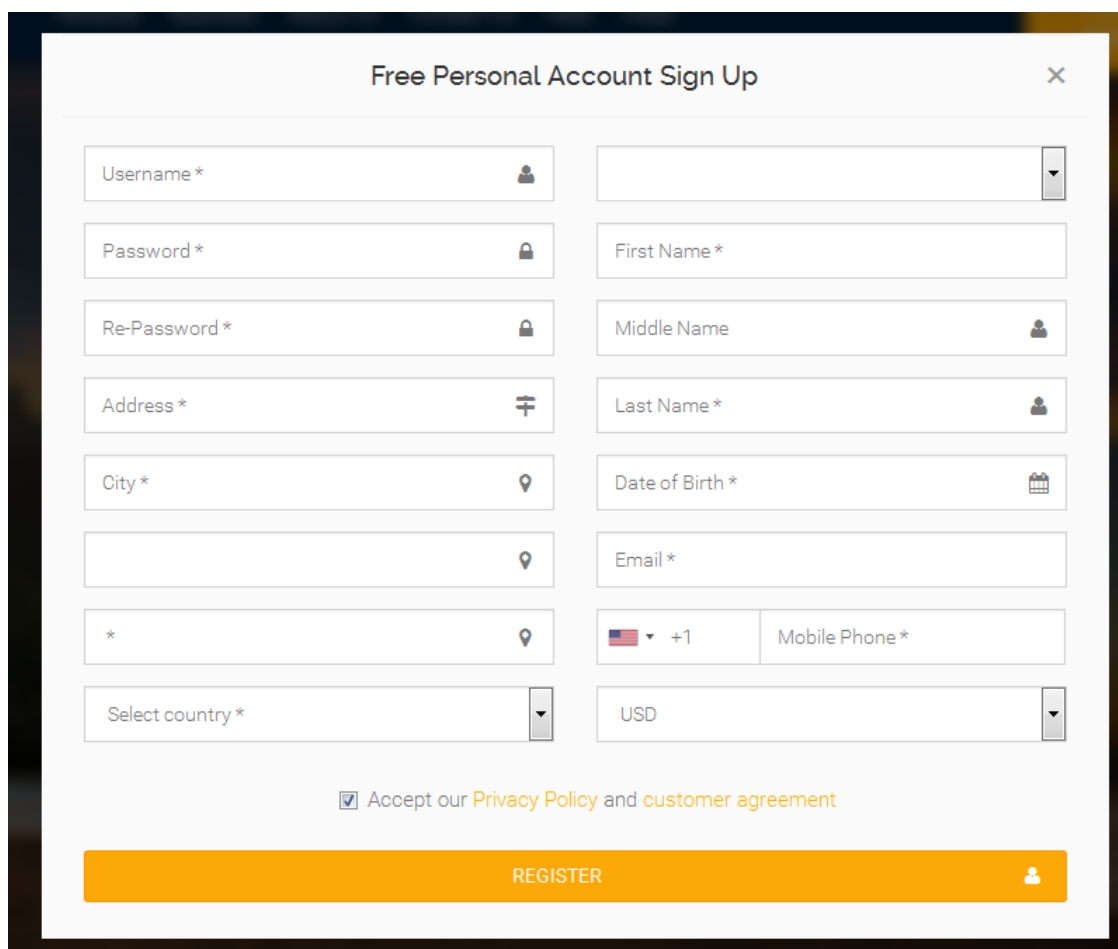
Showing 1 to 1 of 1 entries

« < 1 > »

Figure 4: "My Sponsored Accounts" page

4.2. User registration with merchants referral ID

Merchant provides registration URL with his referral ID to a user. Once user clicks on this link he will be redirected to the UpayCard registration page (see Figure 5: user registration with merchant referral ID).



The image shows a 'Free Personal Account Sign Up' form. It is a two-column layout. The left column contains fields for Username *, Password *, Re-Password *, Address *, City *, a phone number field with a country code dropdown, and a 'Select country *' dropdown. The right column contains a dropdown menu, First Name *, Middle Name, Last Name *, Date of Birth *, Email *, and a Mobile Phone * field with a country code dropdown. Below the form is a checkbox for 'Accept our Privacy Policy and customer agreement' and a large orange 'REGISTER' button with a user icon.


Figure 5: user registration with merchant referral ID

When user fills all required fields and registers, merchant can see registered user as his sponsored account under “My Sponsored Accounts” menu (see Figure 4: “My Sponsored Accounts” page).

5. SPONSORED ACCOUNT CREATION FROM GUI

Merchant can create sponsored accounts from UpayCard. Login to UpayCard as a merchant. Click on the left menu “My Sponsored Accounts”. “My Sponsored Accounts” page will be loaded, which consists of filters (in order to filter when sponsored accounts were created), options for creating sponsored accounts and sponsored accounts list (see Figure 4: “My Sponsored Accounts” page).

In order to create new sponsored account button “Create New Sponsored Account” should be clicked. Then form with fields for new sponsored user creation will be opened (see Figure 6: form for creating new sponsored account).

CREATE SPONSORED ACCOUNT

ACCOUNT DETAILS

Form submission failed!

- You have not answered all required fields
- You have not given a correct date
- You have not given a correct e-mail address

Username: *

Username:

Password: *

Password:

Generate Password

Address

Address

Postal Code

Postal Code

City

City

State

State

Country

Country

First Name *

First/Given Name

Middle name

Middle name

Last Name

Surname/Last Name/Family Name

Date of birth *

Date of birth

Email *

Email

Phone number: *

Gender

Gender

Preferred currency: *

USD

Create Sponsored Account

Figure 6: form for creating new sponsored account

In order to create new sponsored account all required fields needs to be filled and button “Create Sponsored Account” should be clicked. When user is created it should appear in “My Sponsored Accounts” page within sponsored account list (see Figure 4: “My Sponsored Accounts” page).

6. SPONSORED ACCOUNT CREATION USING API

Merchant can create sponsored accounts using UpayCard API.

6.1. CreateUser API Call

Create user via API.

| URL | https://api.upaycard.com/api/merchant/v/1.0/function/create_user |
|-----|---|
|-----|---|

6.1.1. CreateUser Request

| KEY | M | TYPE | LENGTH | DESCRIPTION |
|---------------------------|---|------|--------|--|
| username | Y | AN | 100 | |
| email | Y | AN | 100 | |
| password | Y | AN | 100 | |
| first_name | N | A | 60 | |
| middle_name | N | A | 60 | |
| last_name | N | A | 60 | |
| gender | N | A | 1 | Possible values: F : female M : male |
| bday | N | AN | | ISO 8601 |
| country | N | A | 3 | ISO 3166-1 alpha-3 |
| address_line_1 | N | AN | 100 | |
| address_line_2 | N | AN | 100 | |
| city | N | AN | 45 | |
| state | N | AN | 40 | |
| post_code | N | AN | 10 | |
| billing_country | N | | | ISO 3166-1 alpha-3 |
| billing_address_line_1 | N | | | |
| billing_address_line_2 | N | | | |
| billing_city | N | | | |
| billing_state | N | | | |
| billing_post_code | N | | | |
| preferred_currency | Y | A | 3 | ISO 4217 |
| phone_number | N | AN | 20 | |
| phone_type | N | A | 1 | L = Landline, M = Mobile |
| government_issued_id_type | N | N | 2 | See APPENDIX E: User ID |

| | | | | Types |
|--|---|----|----|---|
| government_issued_id_number | N | AN | 20 | |
| government_issued_id_expiration_date | N | AN | | ISO 8601 |
| government_issued_id_country_of_issuance | N | | | ISO 3166-1 alpha-3 |
| key | Y | AN | 16 | Merchant API Key – provided by UPayCard |
| ts | Y | N | 10 | Request timestamp |
| sign | Y | AN | 32 | See APPENDIX A: SIGN GENERATION |

If first_name, last_name, gender, bday, address_line_1, city, country, post_code, phone_type, phone_number, government_issued_id_type, government_issued_id_number, government_issued_id_expiration_date and government_issued_id_country_of_issuance parameters will be provided, then user profile will be completed.

6.1.2. CreateUser JSON Request Sample

```
{
  "username": "john.doe",
  "email": "john.doe@test.loc",
  "password": "password",
  "first_name": "John",
  "middle_name": null,
  "last_name": "Doe",
  "gender": "M",
  "bday": "1958-08-08",
  "country": "GBR",
  "address1": "",
  "address2": "",
  "city": "London",
  "state": null,
  "post_code": null,
  "billing_country": null,
  "billing_address1": null,
  "billing_address2": null,
  "billing_city": null,
  "billing_state": null,
  "billing_post_code": null,
```

```
"preferred_currency": "USD",  
"key": "_MERCHANT_KEY_",  
"ts": _TIMESTAMP_,  
"sign": "_SIGN_"  
}
```

6.1.3. CreateUser JSON Request Response

| KEY | M | TYPE | LENGHT | DESCRIPTION |
|----------|---|------|--------|---|
| status | Y | AN | 10 | success or error |
| users_id | Y | N | 11 | |
| msg | Y | AN | 255 | See APPENDIX C: Transaction Codes |

6.1.4. CreateUser JSON Response Sample

```
{  
  "status": "success",  
  "users_id": "220",  
  "msg": "Account Created Successfully"  
}
```

6.2. UpdateUserKyc API Call

Upload user's KYC files. Acceptable file types are .jpg, .png, .pdf, .doc.

| | |
|-----|---|
| URL | https://api.upaycard.com/api/merchant/v/1.0/function/update_user_kyc |
|-----|---|

6.2.1. UpdateUserKyc Request

| KEY | M | TYPE | LENGHT | DESCRIPTION |
|-------------------|---|------|--------|---|
| username | Y | AN | 100 | |
| filename | Y | AN | 50 | Must be with file extension |
| doctype | Y | N | 2 | See APPENDIX D: User KYC Document Types |
| organization_name | N | AN | 100 | Mandatory in some cases. See APPENDIX D: User KYC |

| Document Types | | | | |
|--------------------------------|---|----|-----|--|
| file_body | Y | AN | 2MB | base64_encode file |
| notification_url | N | AN | 255 | If provided - URL where notifications will be pushed when KYC status will be changed. Note that POST will be made to the provided URL address. e.g. http://www.mydomain.com/kyc_notifications |
| key | Y | AN | 16 | Merchant API Key – provided by UPayCard |
| ts | Y | N | 10 | Request timestamp |
| sign | Y | AN | 32 | See APPENDIX A: SIGN GENERATION |

6.2.2. UpdateUserKyc JSON Request Sample

```
{
  "username": "rka",
  "filename": "example_passport.jpg",
  "doctype": 1,
  "organization_name": null,
  "file_body": "_BASE_64_ENCODED_FILE_",
  "key": "_MERCHANT_KEY_",
  "ts": "_TIMESTAMP_",
  "sign": "_SIGN_"
}
```

6.2.3. UpdateUserKyc JSON Response

| KEY | M | TYPE | LENGHT | DESCRIPTION |
|---------------|---|------|--------|---|
| status | Y | AN | 10 | success or error |
| msg | Y | AN | 255 | See APPENDIX C: Transaction Codes |

6.2.4. UpdateUserKyc JSON Response Sample

```
{
  "status": "success",
  "msg": "Account owner updated"
}
```

6.3. Programming in PHP

```
// set to 0 in order not to use sandbox
$USE_SANDBOX = 1;

$url_sandbox = "https://sandboxapi.upaycard.com/api/merchant";
$url_live = "https://api.upaycard.com/api/merchant";

$key = "your_key";
$secret = "your_secret";

// adding library provided by UPAYCARD require_once("Api.php");
$url = ($USE_SANDBOX == 1 ? $url_sandbox : $url_live);
$api = new Upaycard_Api($url, $key, $secret);

$username = "joe11"; // username
$email = "joe11@test.com"; // email
$password = "joe11!";
$first_name = "Joe";
$middle_name = ""; //
$last_name = "Dow";
$gender = "M"; // M - male; F - female
$bday = "1995-10-01"; // birthday
$country = "USA"; // ISO 3166-1 alpha-3
$address_line_1 = "Street 1";
$address_line_2 = "";
$city = "My City";
$state = "";
$post_code = "12345";
$billing_country = "";
$billing_address_line_1 = "";
$billing_address_line_2 = "";
$billing_city = "";
$billing_state = "";
$billing_post_code = "";
$preferred_currency = "USD"; // ISO 4217
$phone_number = "";
$phone_type = "M"; // L = Landline, M = Mobile
$government_issued_id_type = "";
$government_issued_id_number = "";
$government_issued_id_expiration_date = ""; // ISO 8601
$government_issued_id_country_of_issuance = ""; // ISO 3166-1 alpha-3

// call createUser API call
$res = $api->createUser($username, $email, $password, $first_name, $middle_name, $last_name,
$gender, $bday, $country, $address_line_1, $address_line_2, $city, $state, $post_code,
$billing_country, $billing_address_line_1, $billing_address_line_2, $billing_city,
$billing_state, $billing_post_code, $preferred_currency, $phone_number, $phone_type,
```

```
$government_issued_id_type, $government_issued_id_number,  
$government_issued_id_expiration_date, $government_issued_id_country_of_issuance);
```

```
if (isset($res['status']) && $res['status'] == 'success') {  
    // do any needed actions after successful user registration  
    // for example upload user KYC  
    $fileName = "my_file_name.jpg";  
    $docType = 4;    // see Appendix D: User kyc document types  
    $org_name = "";  
    $fileContent = base64_encode(file_get_contents('my_file_name.jpg'));  
    // If provided - URL where notifications will be pushed when KYC status will be changed.
```

Note that POST will be made to the provided URL address.

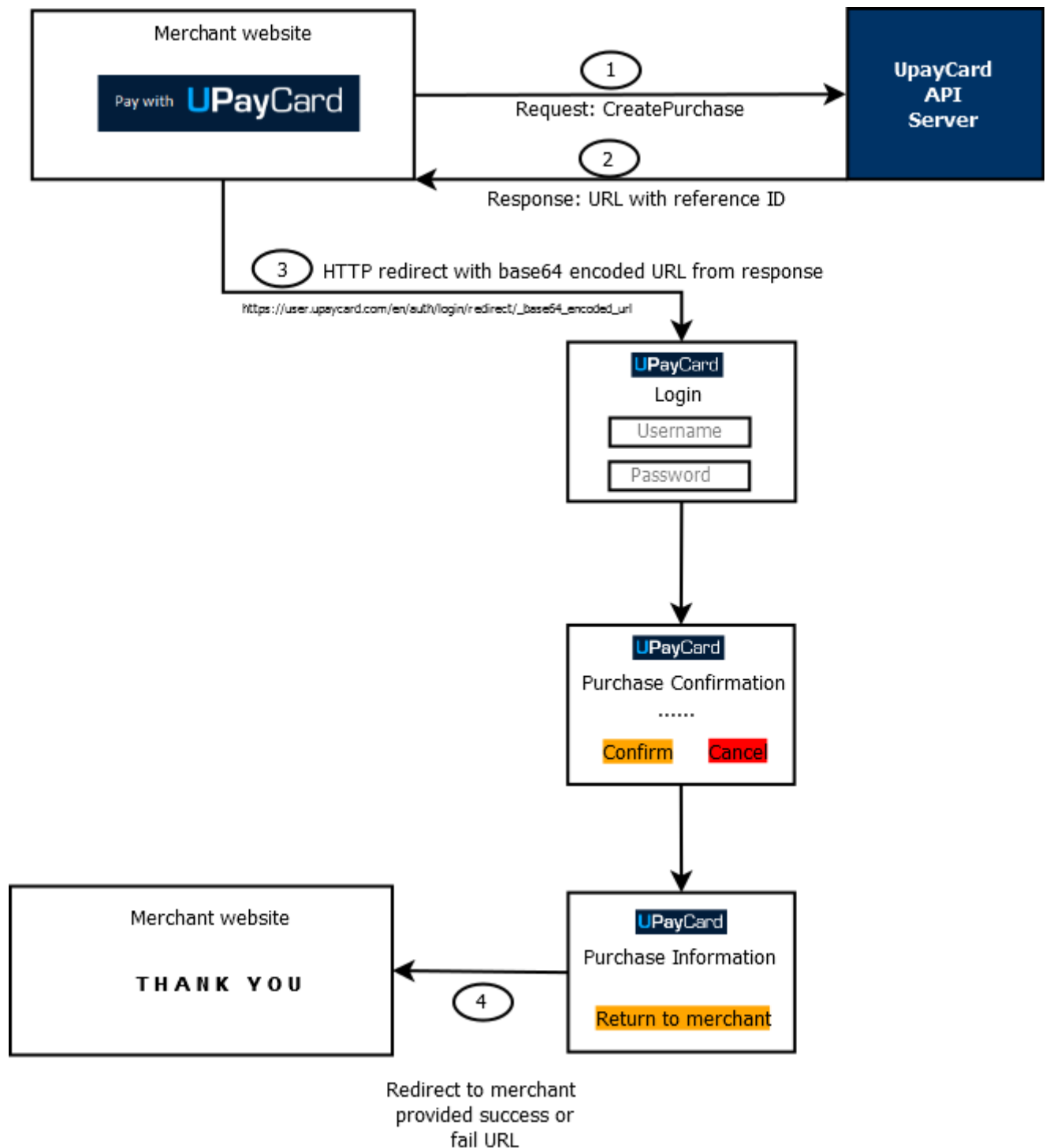
```
    $notification_url = "http://www.mydomain.com/kyc_notifications";  
    $resKYC = $api->updateUserKyc($username, $fileName, $docType, $org_name, $fileContent,  
$notification_url);
```

```
    if (isset($resKYC['status']) && $resKYC['status'] == 'success') {  
        // kyc updated successfully - do needed actions  
    }  
    else {  
        // kyc not updated - do needed actions  
        // to show UpayCard message  
        echo $res['msg'];  
        // otherwise you can show yours message  
    }  
}  
else if (isset($res['msg']) && $res['msg'] == 'error') {  
    // to show UpayCard message  
    echo $res['msg'];  
    // otherwise you can show yours message  
}
```

7. PURCHASE API

7.1. Purchase Flow

The diagram below shows the Checkout API operations that are called during a typical purchase transaction.



Purchase flow step by step:

1. When customer press "Pay with UpayCard" button merchant makes call to UpayCard API function CreatePurchase.
2. In case of success UpayCard API returns data about customer purchase and unique referenceID with URL which merchant will need to encode base64 and redirect to UpayCard.
3. Merchant redirects customer to UpayCard with base64 encoded URL parameter from successful CreatePurchase API function response (example of such URL: https://user.upaycard.com/en/auth/login/redirect/base64_encoded_url). Customer needs to login to UpayCard. After login customer is redirected to "Purchase Information" page (see Figure 7: Purchase Confirmation page) where he can select UpayCard account from which to pay and confirm or cancel his purchase. When customer press either "Confirm" either "Cancel" purchase is being processed (see Figure 8: Purchase Information page - processing). After processing respectively is being showed canceled (see Figure 9: Purchase Information page - cancelled) or completed (see Figure 10: Purchase Information page - completed) purchase.
4. Lastly customer press "Return to merchant" and is redirected to the merchant provided URL (which was provided during CreatePurchase API call).

PURCHASE CONFIRMATION

Sending account *

1778337

Sending account balance *

95120.42 USD

Amount and currency *

8.00 USD

Confirm transfer ✓

Cancel transfer ✕

Figure 7: Purchase Confirmation page

PURCHASE INFORMATION

Processing



Figure 8: Purchase Information page - processing

PURCHASE INFORMATION

Purchase cancelled

[Return to merchant](#)

Figure 9: Purchase Information page - cancelled

PURCHASE INFORMATION

Purchase successfully completed

[Return to merchant](#)

Figure 10: Purchase Information page - completed

7.2. Purchase API calls

7.2.1. CreatePurchase

Initiates purchase procedure.

| | |
|-----|---|
| URL | https://api.upaycard.com/api/merchant/v/1.0/function/create_purchase |
|-----|---|

CreatePurchase Request

| KEY | M | TYPE | LENGHT | DESCRIPTION |
|---------------------|---|------|--------|--|
| receiver_account | Y | N | 11 | Your account ID for receive transfer |
| amount | Y | N | 10,2 | |
| currency | Y | A | 3 | ISO 4217 |
| order_id | Y | A | 50 | Unique order ID |
| sender_user_id | N | N | 11 | Upaycard user ID |
| sender_account | N | N | 11 | Upaycard account ID |
| url_user_on_success | N | A | 255 | URL where user will returned after successful purchase |
| url_user_on_fail | N | A | 255 | URL where user will returned in case of failed purchase |
| url_api_on_success | N | A | 255 | URL where will be sent IPN (instant payment notification) in case of successfully completed purchase. See APPENDIX J: IPN CATCHER PHP Code Example |
| url_api_on_fail | N | A | 255 | URL where will be sent IPN (instant payment notification) in case of failed or canceled purchase. See APPENDIX J: IPN CATCHER PHP Code Example |
| language | N | A | 2 | Preferred language. If not provided – English will be used (only "en" supported at the moment) |
| key | Y | AN | 16 | Merchant API Key – provided by UPayCard |
| ts | Y | N | 10 | Request timestamp |
| sign | Y | AN | 32 | See APPENDIX A: SIGN GENERATION |

CreatePurchase JSON Request Sample

```
{
  "receiver_account": "1000001",
  "amount": "1.01",
  "currency": "USD",
  "order_id": "my_order_81815",
  "sender_user_id": null,
  "sender_account": null,
  "url_user_on_success": "https://www.myeshopexample.com/success_purchase",
  "url_user_on_fail": "https://www.myeshopexample.com/failed_purchase",
  "url_api_on_success": "https://www.myeshopexample.com/ipn_success_purchase",
  "url_api_on_fail": "https://www.myeshopexample.com/ipn_failed_purchase",
}
```

```
"language": "en",
"key": "_MERCHANT_KEY_",
"ts": _TIMESTAMP_,
"sign": "_SIGN_"
}
```

CreatePurchase JSON Response

| KEY | M | TYPE | LENGTH | DESCRIPTION |
|-------------------|---|------|--------|---|
| status | Y | AN | 10 | success or error |
| msg | Y | AN | 255 | |
| reference_id | Y | AN | 16 | Unique ID of the purchase request |
| order_id | Y | A | 50 | Unique order ID |
| amount | Y | N | 10,2 | |
| currency | Y | A | 3 | ISO 4217 |
| purchase_statuses | Y | N | 11 | See APPENDIX H: PURCHASE STATUSES |
| url | Y | AN | 255 | URL where user can finish or cancel his purchase (encode this URL using <i>base64_encode</i> function and redirect to Upaycard login page with additional "reference" param, e.g.: https://user.upaycard.com/en/auth/login/redirect/_base64_encoded_url) |

CreatePurchase JSON Response Sample

```
{
  "status": "success",
  "msg": "Success",
  "reference_id": "ap-57fce78057df6",
  "order_id": "my_order_81815",
  "amount": "1.01",
  "currency": "USD",
  "purchase_status": 1,
  "url": "https://upaycard.com/en/purchase/confirmpurchase/ref/ap-57fce78057df6"
}
```

7.2.2. GetPurchaseStatus

Gets purchase status.

| | |
|-----|---|
| URL | https://api.upaycard.com/api/merchant/v/1.0/function/get_purchase_status |
|-----|---|

GetPurchaseStatus Request

| KEY | M | TYPE | LENGHT | DESCRIPTION |
|--------------|---|------|--------|-----------------------------------|
| reference_id | Y | AN | 255 | Unique ID of the purchase request |

GetPurchaseStatus JSON Request Sample

```
{
  "reference_id": "ap-57fce78057df6"
}
```

GetPurchaseStatus JSON Response

| KEY | M | TYPE | LENGHT | DESCRIPTION |
|-----------------|---|------|--------|---|
| status | Y | AN | 10 | success or error |
| msg | Y | AN | 255 | |
| reference_id | Y | AN | 255 | Unique ID of the purchase request |
| order_id | Y | A | 50 | Unique order ID |
| amount | Y | N | 10,2 | |
| currency | Y | A | 3 | ISO 4217 |
| purchase_status | Y | N | 11 | See APPENDIX H: PURCHASE STATUSES |

GetPurchaseStatus JSON Response Sample

```
{
  "status": "success",
  "msg": "Success",
  "reference_id": "ap-57fce78057df6",
  "order_id": "my_order_81815",
  "amount": "1.01",
  "currency": "USD",
  "purchase_status": 9
}
```

7.3. Programming in PHP

```
// =====
// 1 STEP: creating purchase request
// =====
```

```
// set to 0 in order not to use sandbox
$USE_SANDBOX = 1;

$url_sandbox = "https://sandboxapi.upaycard.com/api/merchant";
$url_live = "https://api.upaycard.com/api/merchant";

$redirect_sandbox = "https://sandboxauser.upaycard.com/en/auth/login/redirect";
$redirect_live = "https://user.upaycard.com/en/auth/login/redirect/";

$key = "your_key";
$secret = "your_secret";

// adding library provided by UPAYCARD require_once("Api.php");
$url = ($USE_SANDBOX == 1 ? $url_sandbox : $url_live);
$api = new Upaycard_Api($url, $key, $secret);

$receiver_account = ""; // your receiving_account number
$sender = ""; // sender account id/username/email
$amount = "100.00";
$currency = "USD"; // currency code (USD or EUR)
$order_id = ""; // order_id in your system
$sender_user_id = null; // should be NULL as you do not know sender user ID
$sender_account_id = null; // should be NULL as you do not know sender account ID
$url_user_on_success = 'http://www.myeshop.com/success'; // where user will be redirected
after successfull purchase
$url_user_on_fail = 'http://www.myeshop.com/fail'; // where user will be redirected after
successfull purchase
// URL where will be sent IPN (instant payment notification) in case of successfully
completed purchase
$url_api_on_success = 'http://www.myeshop.com/ipn_success';
// URL where will be sent IPN (instant payment notification) in case of failed or canceled
purchase
$url_api_on_fail = 'http://www.myeshop.com/ipn_fail';
$language = 'en'; // only 'en' is supported at the moment

// initialize purchase and get URL for redirection
$res = $api->createPurchase($receiver_account, $amount, $currency, $order_id,
$sender_user_id, $sender_account_id, $url_user_on_success, $url_user_on_fail,
$url_api_on_success, $url_api_on_fail, $language);

if (isset($res['status']) && $res['status'] == 'success') {
    $referenceID = (isset($res['reference_id']) ? $res['reference_id'] : null);

    if (isset($res['url']) && !empty($res['url'])) {
        $url_redirect = ($USE_SANDBOX == 1 ? $redirect_sandbox : $redirect_live);
        $redirect = $url_redirect . base64_encode($res['url']);
        header("Location: " . $redirect);
    }
    else {
```

```
        // someting went wrong - show error message
    }
}
else {
    if (isset($res['msg']) && $res['msg'] == 'error') {
        // to show UpayCard message
        echo $res['msg'];
        // otherwise you can show yours message
    }
    else {
        // case which should never happen, but be prepare for it too
        // show some error message
    }
}

// =====
// 4 STEP: customer is returned to the merchant website
// =====
// when customer returns to merchant website
// merchant can always check purchase status
// and show message respectively

$resStatus = $api->getPurchaseStatus($referenceID);

if (isset($resStatus['status']) && $resStatus['status'] == 'success') {
    // possible purchase status:
    // 1 - Created
    // 2 - Logged in
    // 4 - Processing
    // 7 - Canceled
    // 8 - Failed
    // 9 - Successful

    switch ($resStatus['purchase_status']) {
        case 9:
            echo 'Thank you for your purchase.';
            break;
        case 4:
            echo 'Your purchase is processing...'; // implement ajax call to check status
            break;
        // ...
        default:
            break;
    }
}
}
```

8. PURCHASE URL

8.1. Purchase Flow

Purchase flow step by step:

1. Customer comes to the Merchant payments options page (see Figure 11: merchant page with payment options) and chooses to pay with UpayCard.

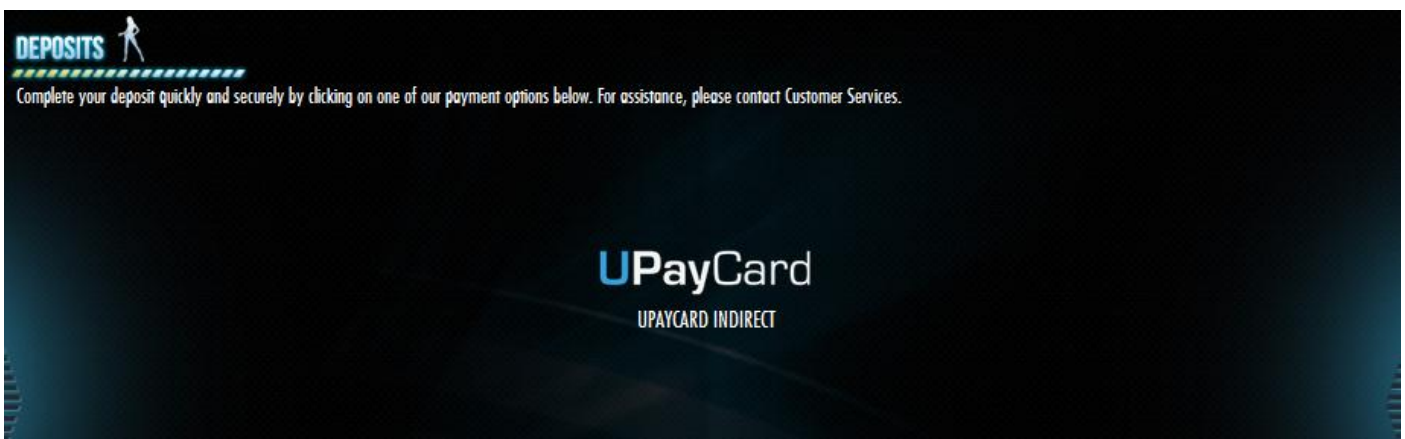


Figure 11: merchant page with payment options

2. When customer press “Pay” button (see Figure 12: pay with UpayCard) merchant collects all needed data, encodes it and redirects user to UpayCard (e.g. https://upaycard.com/en/purchase/initiatepurchase/data/_base64_encoded_data_)



Figure 12: pay with UpayCard

3. In case of successfully initialised purchase UpayCard redirects user to UpayCard login screen (see Figure 13: UpayCard login page). Customer needs to login to UpayCard. After login customer is redirected to "Purchase Information" page (see Figure 7: Purchase Confirmation page) where he can select UpayCard account from which to pay and confirm or cancel his purchase. When customer press either "Confirm" either "Cancel" purchase is being processed (see Figure 8: Purchase Information page - processing). After processing respectively is being showed canceled (see Figure 9: Purchase Information page - cancelled) or completed (see Figure 10: Purchase Information page - completed) purchase.

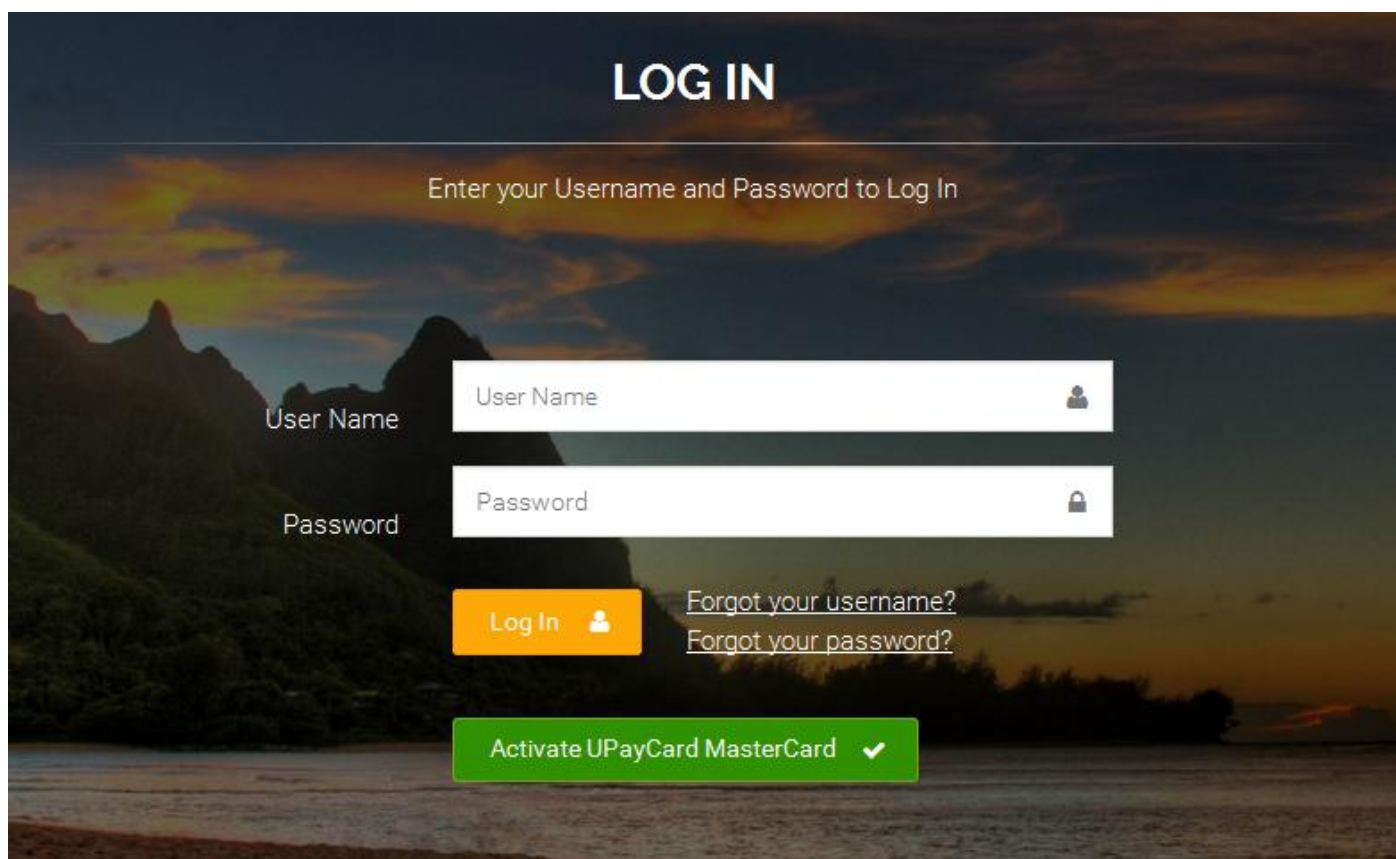


Figure 13: UpayCard login page

4. Lastly customer press "Return to merchant" and is redirected to the merchant provided URL (which was provided in encoded parameters during request in first step).
5. If merchant provided "url_api_on_success" and "url_api_on_fail" he will receive IPN after purchase was failed or completed. Note that UpayCard tries to send IPN 10 times. After each failed time, next IPN sending is delayed for some time. See 17 APPENDIX J: IPN CATCHER PHP Code Example.

8.2. Programming in PHP

Example below provides Purchase Form for testing. Merchant should implement its form as need, then after user press button (after POST) collect all data, encode it and redirect to UpayCard.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Purchase Test Form</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <style>
      .required {
        color: red;
      }
      input { width: 300px; }
    </style>
  </head>
  <body>
    <h1>Purchase Test Form</h1>

    <form method="post" action="">
      <table>
        <tr>
          <td>Receiver account <span class="required">*</span></td>
          <td><input name="receiver_account" value="" placeholder="Your
account ID" required /></td>
        </tr>
        <tr>
          <td>Amount <span class="required">*</span></td>
          <td><input name="amount" value="" placeholder="amount"
required /></td>
        </tr>
        <tr>
          <td>Currency <span class="required">*</span></td>
          <td><input name="currency" value="" placeholder="USD or EUR"
required /></td>
        </tr>
        <tr>
          <td>Order ID <span class="required">*</span></td>
          <td><input name="order_id" value="php echo
uniqid('orderId_'); ?" required /></td>
        </tr>
        <tr>
          <td>Success URL <span class="required">*</span></td>
          <td><input name="url_user_on_success" value=""
placeholder="http://www.mysite.com/success_page_for_user" required /></td>
        </tr>
      </table>
    </form>
  </body>
</html>
```

```

        <tr>
            <td>Fail URL <span class="required">*</span></td>
            <td><input name="url_user_on_fail" value=""
placeholder="http://www.mysite.com/failed_page_for_user" required /></td>
        </tr>
        <tr>
            <td></td>
            <td><input type="submit" value="Submit" /> </td>
        </tr>
    </table>
</form>
</body>
</html>

<?php
    $USE_SANDBOX = 1;

    $url_sandbox =
"https://sandboxauser.upaycard.com/en/purchase/initiatepurchase/?data=";
    $url_live = "https://upaycard.com/en/purchase/initiatepurchase/?data=";

    if (isset($_POST) && !empty($_POST)) {
        // create hash add it to the params
        $strToSign = $_POST['receiver_account'] . $_POST['amount'] .
$_POST['order_id'] . $_POST['currency'];
        $hash = md5($strToSign);
        $_POST['hash'] = $hash;

        // encoded params
        $encoded = base64_encode(json_encode($_POST));
        $url = ($USE_SANDBOX == 1 ? $url_sandbox : $url_live) .
urlencode($encoded);


        // redirect user
        header("Location:" . $url);
    }
?>

```

8.3. URL settings for IPN

URLs for success or failed IPN messages can be set in “My Profile” under “API” tab (see Figure 14: API URLs page). There are two inputs on the page:

- URL for successfully finished purchases
- URL for failed purchases



MY PROFILE

User Details

Addresses

Security

Documents

Bank Details

Websites

Credit/Debit Cards

API

UPDATE API URLS

URL API on success *

URL API on fail *

Update




Figure 14: API URLS page

9. APPENDIX A: SIGN GENERATION

Sign of request, it is MD5 hash of keys, values and secret.

Example of Sign generation:

```
MD5("key1:value1: key2: value2:...:key:_MERCHANT_KEY_:ts:_TIMESTAMP_:_SECRET_")
```

Example in PHP:

```
function _sign($params)
{
    $strToSign = '';
    $params['key'] = '_MERCHANT_KEY_';
    $params['ts'] = time();
    foreach ($params as $k => $v)
        if($v !== NULL)
            $strToSign .= "$k:$v:";
    $strToSign .= '_MERCHANT_SECRET_';

    $params['sign'] = md5($strToSign);
    return $params;
}
```

10. APPENDIX B: ENCRYPT/DECRYPT INFORMATION

Some information in response (like credit card number) will be encrypted so must use decryption function to get real information. For decrypt must have MERCHANT_3DES_KEY.

Decryption function in PHP:

```
function decrypt3DES($encrypted) {
    $len = strlen(_MERCHANT_3DES_KEY_);
    $key = $len < 24 ? _MERCHANT_3DES_KEY_ . substr(_MERCHANT_3DES_KEY_, 0, 24 - $len) : _MERCHANT_3DES_KEY_;
    $encrypted = base64_decode( $encrypted );
    $out = mcrypt_decrypt(MCRYPT_3DES, $key, $encrypted, MCRYPT_MODE_CBC, substr(_MERCHANT_3DES_KEY_, 0, 8));
    $block = mcrypt_get_block_size('tripledes', 'cbc');
    $packing = ord($out{strlen($out) - 1});
    if ($packing && ($packing < $block)) {
        for ($P = strlen($out) - 1; $P >= strlen($out) - $packing; $P--) {
            if (ord($out{$P}) != $packing) {
```

```
        $packing = 0;
    }
}
}
$out = substr($out,0,strlen($out) - $packing);
return $out;
}

function encrypt3DES($data) {
    $len = strlen(_MERCHANT_3DES_KEY_);
    $key = $len < 24 ? _MERCHANT_3DES_KEY_ . substr(_MERCHANT_3DES_KEY_, 0, 24 -
    $len) : _MERCHANT_3DES_KEY_;

    $l = 8 - strlen($data) % 8;
    if ($l > 0)
        $data .= str_repeat(chr($l), $l);

    $out = mcrypt_encrypt(MCRYPT_3DES, $key, $data, MCRYPT_MODE_CBC,
    substr(_MERCHANT_3DES_KEY_, 0, 8));

    return base64_encode($out);
}
```

11. APPENDIX C: TRANSACTION CODES

| Code | Description |
|------|--|
| 000 | Transaction successfully completed |
| 100 | Load limit exceeded (value of transactions) |
| 101 | Load limit exceeded (number of transactions) |
| 102 | Transfer limit exceeded (maximum transaction amount allowed) |
| 103 | Transfer limit exceeded (value of transactions) |
| 104 | Transfer limit exceeded (number of transactions) |
| 105 | Withdrawal limit exceeded (value of transactions) |
| 106 | Withdrawal limit exceeded (number of transactions) |
| 107 | Withdrawal limit exceeded (maximum transaction amount allowed) |
| 108 | Card throughput limit exceeded (must provide KYC documents) |
| 110 | Transfer restricted |
| 111 | Load restricted |
| 112 | Recipient cannot accept transfers |
| 113 | Account balance exceeded |
| 114 | Operation is not allowed |
| 200 | Insufficient funds |
| 300 | Card is inactive |
| 400 | Could not find currency rate. |

| | |
|-----|--|
| 500 | Invalid signature |
| 501 | Error creating session |
| 502 | Operation is not allowed |
| 503 | Missing field |
| 504 | Field format error |
| 505 | Invalid receiver account |
| 506 | User not found |
| 507 | Invalid currency code |
| 508 | Invalid sender account |
| 509 | Define sender account |
| 510 | Duplicate order id |
| 511 | Initialized transaction not found |
| 512 | Wrong Key Code provided |
| 513 | Transfer request already confirmed |
| 514 | Transaction not found |
| 515 | Transaction cannot be refunded |
| 516 | Cannot refund this amount |
| 517 | Your transaction request was sent to our Bank for processing |
| 999 | Unknown error |

12. APPENDIX D: USER KYC DOCUMENT TYPES

| Type ID | Description | Mandatory to provide company name |
|---------|-------------------------------|-----------------------------------|
| 4 | Passport | N |
| 5 | Drivers License Front | N |
| 6 | State/Province ID Front | N |
| 7 | Utility Bill | Y |
| 8 | Bank Statement | N |
| 9 | Credit/Debit Card Front | N |
| 10 | ACH Voided Check | N |
| 11 | Check21 | N |
| 12 | Marriage Certificate | N |
| 13 | Power of Attorney | N |
| 14 | Subpoena | N |
| 15 | Marriage Dissolution | N |
| 16 | Notarized Statement | N |
| 17 | Document of Legal Name Change | N |
| 18 | Police Report | N |
| 19 | Disputes | N |
| 20 | Wire Transfers | N |

| | | |
|----|---|---|
| 21 | Limit Increase | N |
| 22 | Email Change | N |
| 23 | ACH Demand Draft | N |
| 24 | Application Form | N |
| 26 | Contract | N |
| 28 | Incorporation of Company | N |
| 30 | Memorandum and Articles | Y |
| 32 | Shareholder Documentation | N |
| 34 | Proof of Company Address | N |
| 36 | Other | N |
| 38 | Company registration documents | N |
| 40 | UPayCard Statement | N |
| 42 | Credit/Debit Card Back | N |
| 44 | Business Proposal | N |
| 46 | Business Agreement | N |
| 48 | Agent Agreement | N |
| 50 | Driver's License Back (If applicable) | N |
| 52 | Passport-Additional Pages (If applicable) | N |
| 54 | State/Province ID Back (If applicable) | N |
| 56 | Government-issued ID | N |
| 58 | Personal Identification Card | N |
| 60 | Residence Permit | N |
| 62 | Screenshot | N |

13. APPENDIX E: USER ID TYPES

| Type ID | Description |
|---------|------------------------------------|
| 1 | Passport Registry No. |
| 2 | Personal Identification No. |
| 3 | Identity Card No. |
| 4 | Utility Bill |
| 8 | Travel Document |
| 12 | Residence Permit |
| 13 | Identity Certificate No. |
| 16 | Registro Federal de Contribuyentes |
| 17 | Credencial de Elector |
| 19 | Social Security Number (US ONLY) |
| 20 | Tax File Number (Australia Only) |

14. APPENDIX F: PHP CODE EXAMPLE

```
function _request($servicename, $params)
{
    ini_set('max_execution_time', 300);

    $suri = '_SERVICE_URL_' . '/v/' . '_VERSION_' . '/function/' .
'_API_FUNCTION_NAME_' ;

    $sstr = json_encode($params);

    $ch = curl_init( $suri );
    curl_setopt($ch, CURLOPT_CUSTOMREQUEST, "POST");
    curl_setopt($ch, CURLOPT_POSTFIELDS, $sstr);
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
    curl_setopt($ch, CURLOPT_TIMEOUT, 300);
    curl_setopt($ch, CURLOPT_HTTPHEADER, [
        'Content-Type: application/json',
        'Content-Length: ' . strlen($sstr)
    ]);
    $response = curl_exec($ch);
    curl_close($ch);

    return $response;
}
```

15. APPENDIX G: Invalid requests error codes

| Type ID | Description |
|---------|--|
| 1001 | API version parameter 'v' is not provided |
| 1002 | Wrong API version provided |
| 1003 | Function name parameter 'function' is not provided |
| 1004 | Wrong method name provided |
| 1005 | No data received |
| 1006 | Required field is not provided or is empty |
| 1007 | System error |

16. APPENDIX H: Purchase statuses

| Type ID | Description |
|---------|-------------|
| 1 | Created |
| 2 | Logged in |
| 4 | Processing |
| 7 | Canceled |

| | |
|---|-------------|
| 8 | Failed |
| 9 | Successfull |

17. APPENDIX J: IPN CATCHER PHP Code Example

```
if (!empty($_POST)) {
    $status_id = $_POST['status_id'];           // 7
    $reference_id = $_POST['reference_id'];      // 'ap-57fcb50701182'
    $order_id = $_POST['order_id'];             // 'my_order_81815'
    $data_amount = $_POST['data_amount'];       // 1.01
    $currency = $_POST['currency'];             // 'USD'
    $transaction_id = $_POST['transaction_id'];  // transaction ID

    // note that transaction ID will be not empty only if status is 8 or 9

    if (!empty($_POST['sign'])) {
        $sign = $_POST['sign'];
        $arrToHash = [];

        foreach ($_POST as $field => $value) {
            if ($field == 'sign' || $value === null) { continue; }
            $arrToHash[] = $field . ':' . $value;
        }

        $hash = md5(implode(':', $arrToHash) . ':' . '_MERCHANT_SECRET_');
        if ($hash == $sign) {
            // sign correct - any needed actions further
        }
        else {
            // wrong sign - any needed actions further
        }
    }
    else {
        // no sign - any needed actions further
    }
}
```