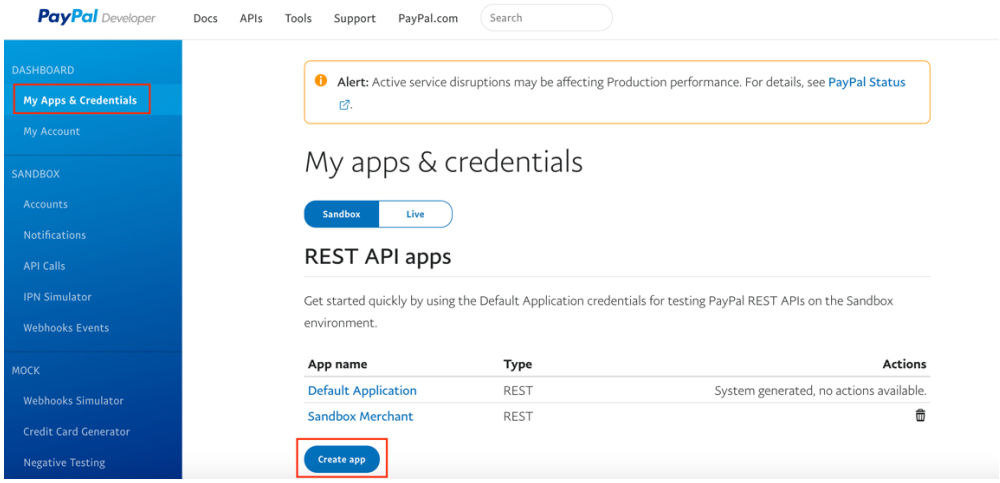
PayPal 信用卡 (Advance Credit Card) 集成方案

这份文档会提供关如何集成 Advance Credit Card (ACC)付款以及 3Ds 详细资料。ACC 一般支持最新的 SDK (Restful API) 版本去進行付款版本去进行付款，如果你现在使用旧版本的 Classic API (NVP/SOAP)，可能需要去更新你的 API 使用版本，而你也可以参考这份官方文档。

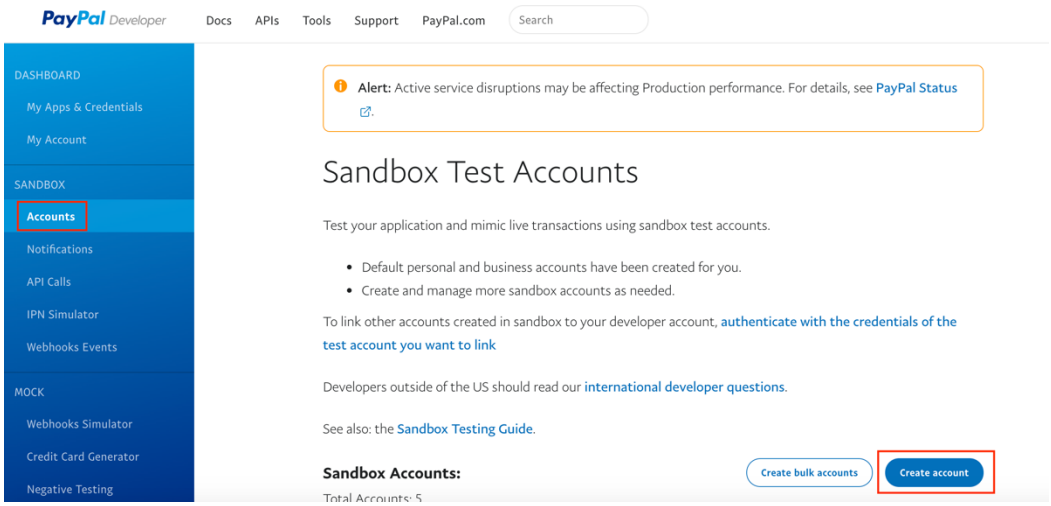
帐号设定

在集成解决方案前，请先准备以下资料：

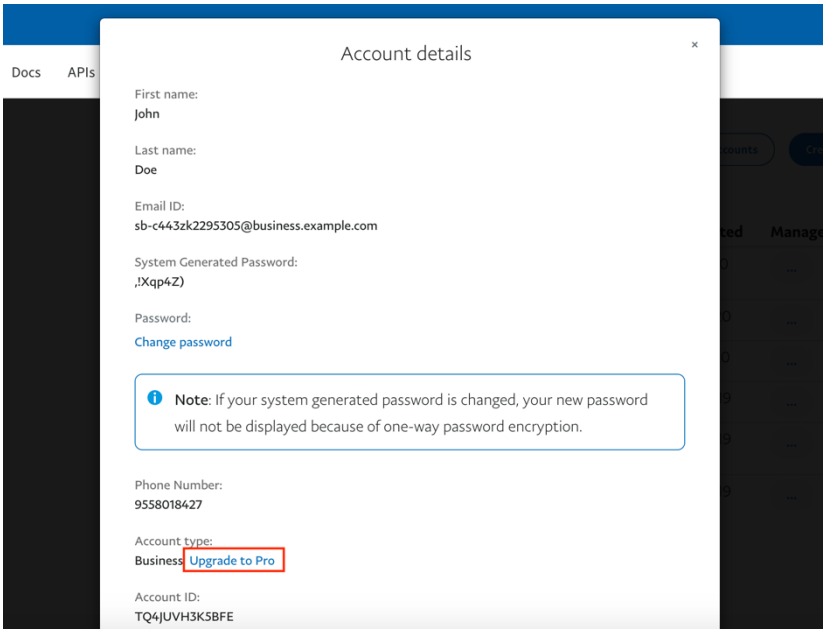
1. 从沙箱帐号中建立一个新的应用程式：
   进入 PayPal 开发人员网页，登入你的 PayPal 沙箱帐号（香港）。从左边选项中点选 My Apps & Credentials，再点选 Create App，输入自定义的应用程式名称，然後建立一个新的应用程式。



2. 商业帐户启动 Payment-Pro 模式：
   从左边选项中点选 Accounts，再点选 Create Account 建立一个新的商业帐户（点选合适地區）。



建立帐户后编辑帐户资料，然后点选 Upgrade to Pro。



UCC 集成内容

在这个信用卡付款方案上，我们只需要用户提交信用卡号码，信用卡有效日期及 CVV 去完成付款。你们可以自定义帐单地址，然后把资料加到编码里面。

1. 从 API credentials 中获取 access token：
   你可以从你的应用程式中的 client ID 和 secret 去發 HTTP 請求去获取 access token. 这份文档说明了如何用 Postman 去设定 API 请求格式。

```bash
curl -v POST https://api.sandbox.paypal.com/v1/oauth2/token \
  -H "Accept: application/json" \
  -H "Accept-Language: en_US" \
  -u "CLIENT_ID:SECRET" \
  -d "grant_type=client_credentials"
```

API 沙箱环境 URL: https://api.sandbox.paypal.com/v1/oauth2/token
API 真实环境 URL: https://api.paypal.com/v1/oauth2/token

你也可以从 Postman 中的设定和請求結果去获取 “access_token” 资料

2. 获取 client data token 资料:
   拿到了 access token 后, JavaScript SDK 需要用戶提供 client data token 的资料。以下是有关 HTTP 请求裡 API 的詳細資料：

```bash
curl -X POST https://api.sandbox.paypal.com/v1/identity/generate-token \
-H 'Content-Type: application/json' \
-H 'Authorization: Bearer <ACCESS-TOKEN>' \
-H 'Accept-Language: en_US' \
-d '{
 "customer_id": "customer_1234"
}'
```

API 沙箱环境 URL: https://api.sandbox.paypal.com/v1/identity/generate-token
API 真实环境 URL: https://api.paypal.com/v1/identity/generate-token

你可以自定义你的 customer_id 资料，但你需要确定数值是独立单一，而且同一用户不可以改变数值。

3. 把 PayPal Javascript SDK 和信用卡列表加進你的网頁
   你可以把 PayPal JavaScript SDK 和信用卡列表加進你的购物车网頁。以下的例子是有关如何将信用卡列表放进编码。这份文档是 cardfields.css 的原码档。

   在页面显示之前，你可能需要在你的後台去创建一个新的付款要求去获取 order ID。在这之后，请把 order ID 放在 SDK 编码上。如果你想知道如何建立一个新的付款请求，请往这里 。

你也可以参考这个演示例子。

```
<html>
<head>

 <meta charset="utf-8"/>
 <meta name="viewport" content="width=device-width, initial-scale=1"> <!-- Optimal rendering on mobile devices. -->
 <meta http-equiv="X-UA-Compatible" content="IE=edge" /> <!-- Optimal Internet Explorer compatibility -->
 <script src="http://code.jquery.com/jquery-1.11.1.min.js"></script>
 <link rel="stylesheet" type="text/css" href="cardfields.css"/>

</head>
<body>

<!-- JavaScript SDK -->
 <script src="https://www.paypal.com/sdk/js?components=hosted-fields,buttons&client-id=<YOUR-CLIENT-ID>" data-client-token="<YOUR-CLIENT-TOKEN>"></script>

  <!-- Advanced credit and debit card payments form -->
  <div class='card_container'>
   <form id='my-sample-form'>
    <label for='card-number'>Card Number</label><div id='card-number' class='card_field'></div>
    <div>
     <label for='expiration-date'>Expiration Date</label><div id='expiration-date' class='card_field'></div>
    </div>
    <div>
     <label for='cvv'>CVV</label><div id='cvv' class='card_field'></div>
    </div>
    <label for='card-holder-name'>Name on Card</label><input type='text' id='card-holder-name' name='card-holder-name' autocomplete='off' placeholder='card holder name'/>
    <div>
     <label for='card-billing-address-street'>Billing Address</label><input type='text' id='card-billing-address-street' name='card-billing-address-street' autocomplete='off' placeholder='street address'/>
    </div>
    <div>
     <label for='card-billing-address-unit'>&amp;nbsp;</label><input type='text' id='card-billing-address-unit' name='card-billing-address-unit' autocomplete='off' placeholder='unit'/>
    </div>
    <div>
     <input type='text' id='card-billing-address-city' name='card-billing-address-city' autocomplete='off' placeholder='city'/>
    </div>
    <div>
     <input type='text' id='card-billing-address-state' name='card-billing-address-state' autocomplete='off' placeholder='state'/>
    </div>
    <div>
     <input type='text' id='card-billing-address-zip' name='card-billing-address-zip' autocomplete='off' placeholder='zip / postal code'/>
    </div>
    <div>
     <input type='text' id='card-billing-address-country' name='card-billing-address-country' autocomplete='off' placeholder='country code' />
    </div>
```

```
    <br><br>
    <button value='submit' id='submit' class='btn'>Pay</button>
   </form>
  </div>

  <!-- Implementation -->
  <script>
   // Eligibility check for advanced credit and debit card payments
   if (paypal.HostedFields.isEligible()) {
    paypal.HostedFields.render({
     createOrder: function () {return "order-ID";}, // replace order-ID with the order ID
     styles: {
      'input': {
       'font-size': '17px',
       'font-family': 'helvetica, tahoma, calibri, sans-serif',
       'color': '#3a3a3a'
      },
      ':focus': {
       'color': 'black'
      }
     },
     fields: {
      number: {
       selector: '#card-number',
       placeholder: 'card number'
      },
      cvv: {
       selector: '#cvv',
       placeholder: 'card security number'
      },
      expirationDate: {
       selector: '#expiration-date',
       placeholder: 'mm/yy'
      }
     }
    }).then(function (hf) {
     $('#my-sample-form').submit(function (event) {
      event.preventDefault();
      hf.submit({
       // Cardholder Name
       cardholderName: document.getElementById('card-holder-name').value,
       // Billing Address
       billingAddress: {
        streetAddress: document.getElementById('card-billing-address-street').value,     // address_line_1 -
street
        extendedAddress: document.getElementById('card-billing-address-unit').value,      // address_line_2 -
unit
        region: document.getElementById('card-billing-address-state').value,          // admin_area_1 - state
        locality: document.getElementById('card-billing-address-city').value,         // admin_area_2 - town / city
        postalCode: document.getElementById('card-billing-address-zip').value,         // postal_code -
postal_code
        countryCodeAlpha2: document.getElementById('card-billing-address-country').value   // country_code -
country
       }
      }).then((result) => {
```

```
            //TODO
            // Fetch the API call from backend to capture the order
          });
        });
      });
    }
    else {
      $('#my-sample-form').hide();  // hides the advanced credit and debit card payments fields if merchant isn't
eligible
    }
  </script>

  </body>
  </html>
```

4. 从后台获取用户的付款要求
   当信用卡资料发送以后，你可以从后台发送 HTTP 请求去获取用户付款请求。
   你可以把 API 回调资料发送到购物车页面。

   以下编码是有关如何获取用户的付款请求：

```
// Note: This is example code. Each server platform and programming language has a different way of
handling requests, making HTTP API calls, and serving responses to the browser.

// 1. Set up your server to make calls to PayPal

// 1a. Add your client ID and secret
PAYPAL_CLIENT = 'PAYPAL_SANDBOX_CLIENT';
PAYPAL_SECRET = 'PAYPAL_SANDBOX_SECRET';

// 1b. Point your server to the PayPal API
PAYPAL_OAUTH_API = 'https://api.sandbox.paypal.com/v1/oauth2/token/';
PAYPAL_ORDER_API = 'https://api.sandbox.paypal.com/v2/checkout/orders/';

// 1c. Get an access token from the PayPal API
basicAuth = base64encode(`${ PAYPAL_CLIENT }:${ PAYPAL_SECRET }`);
auth = http.post(PAYPAL_OAUTH_API {
  headers: {
    Accept:       `application/json`,
    Authorization: `Basic ${ basicAuth }`
  },
  data: `grant_type=client_credentials`
});

// 2. Set up your server to receive a call from the client
function handleRequest(request, response) {

  // 2a. Get the order ID from the request body
  orderID = request.body.orderID;

  // 3. Call PayPal to capture the order
```

```
  capture = http.post(PAYPAL_ORDER_API + orderID + '/capture', {
    headers: {
      Accept:       `application/json`,
      Authorization: `Bearer ${ auth.access_token }`
    }
  });

  // 4. Save the capture ID to your database
  if (!capture.error) {
    captureID = capture.purchase_units[0]
        .payments.captures[0].id;
    database.saveCaptureID(captureID);
  }

  // 5. Handle any errors from the call
  if (capture.error) {
    console.error(capture.error);
    return response.send(500);
  }

  // 6. Return a successful response to the client
  response.send(200);
}
```

从返回结果，你可以获取正确的付款状态：

```
"result": {
      "id": "6N283131LV7966709",
      "intent": "CAPTURE",
      "purchase_units": [
       {
        "reference_id": "666a97cddb3245789e23d3299ad9e7a0",
        "amount": {
         "currency_code": "USD",
         "value": "2.00"
        },
        "payee": {
         "email_address": "xxx@xxx.com”,
         "merchant_id": "P2FFKES9RYGKJ"
        },
        "description": "xxxxxxxx",
        "custom_id": "xxxxxx",
        "soft_descriptor": "PAYPAL *PEPPER HK",
        "payments": {
         "captures": [
          {
           "id": "0CX03331WX398261F",
           "status": "COMPLETED",
           "amount": {
            "currency_code": "USD",
            "value": "2.00"
           },
           "final_capture": true,
           "seller_protection": {
```

```
            "status": "ELIGIBLE",
            "dispute_categories": [
              "ITEM_NOT_RECEIVED",
              "UNAUTHORIZED_TRANSACTION"
            ]
          },
          "custom_id": "xxxxxx",
          "links": [
            {
              "href": "https://api.paypal.com/v2/payments/captures/0CX03331WX398261F",
              "rel": "self",
              "method": "GET"
            },


.....
```


3D-S 集成方案

3D Secure 是由发卡银行授权给信用卡持有者，从而去解决恶意退款或欺诈等问题。當 3D Secure 授權后，它可以把恶意退款的賠償問題由商戶轉向給發卡銀行去負責，或者你可以参考这份文档 。以下例子是有关集成方法：

1. 输入参数去启动 3D Secure:
   在 SDK script tag 加入参数 data-enable-3ds：

```html
<script src="https://paypal.com/sdk/js?client-id=YOUR_CLIENT_ID" data-client-token="eyJicmFpbnRyZWUiOnsiYXV0aG9ya==" data-enable-3ds></script>
```

2. 在 3D Secure 編碼上加入 contingency for:
   在购物车上加入以下編碼:

```
<div id="payments-sdk__contingency-lightbox"></div>
```

3. 更新 UCC 編碼:
   你可能需要加入 contingency 参数去提交授權 3D Secure：

```
```

```
// Check eligibility for advanced credit and debit card payments
if (paypal.HostedFields.isEligible()) {
  // render the card fields
  paypal.HostedFields.render({

    // sample function to return the order ID
    createOrder: () => {
      // add logic to return an order ID from your server
    },
    fields: {
      number: {
        selector: '#card-number',
        placeholder: 'card number'
      },
      cvv: {
        selector: '#cvv',
        placeholder: 'CVV',
      },
      expirationDate: {
        selector: '#expiration-date',
        placeholder: 'mm/yyyy'
      }
    }
  }).then(function (hf) {

    document.querySelector('#my-sample-form').addEventListener('submit', (event) => {
      event.preventDefault();

      hf.submit({

        // Trigger 3D Secure authentication
        contingencies: ['3D_SECURE']

      }).then(function (payload) {

        /** sample payload
        * {
        * "orderId": "0BS14434UR665304G",
        * "liabilityShifted":      true,
        * "authenticationStatus":   "YES",
        * "authenticationReason":   "SUCCESSFUL"
        * }
        * possible value:
        * liabilityShifted - true, false, undefined
        * authenticationStatus - "YES", "NO", "ERROR", undefined
        * authenticationReason - "SUCCESSFUL", "ATTEMPTED", "BYPASSED", "UNAVAILABLE", "ERROR",
"CARD_INELIGIBLE", "SKIPPED_BY_BUYER", "FAILURE", undefined
        */

        // Needed only when 3D Secure contingency applied

        if (payload.liabilityShifted === undefined) {
          // Handle no 3D Secure contingency passed scenario
        }

        if (payload.liabilityShifted) {
```

```
            // Handle Buyer confirmed 3D Secure successfully
        }

        if (payload.authenticationReason === 'SKIPPED_BY_BUYER') {
            // Handle buyer skipped 3D Secure use-case
        }
        });
      });
    });
}
else {
  /*
   * Handle experience when advanced credit and debit card payments
   * card fields are not eligible
   */
}
```

在返回結果裡，会在 liabilityShifted， authenticationStatus 和 authenticationReason 显示不同的授权结果如下：

Liability shifted	Authentication Status	Authentication Reason	Description	Next Steps
undefined	undefined	undefined	You have not required 3D Secure for the buyer or the card network did not require a 3D Secure	You can continue with authorization and assume liability. If you prefer not to assume liability, ask the buyer for another card
true	YES	SUCCESSFUL	Buyer successfully authenticated using 3D secure	Buyer authenticated with 3DS and you can continue with the authorization
false	ERROR	ERROR	An error occurred with the 3D Secure authentication system	Prompt the buyer to re-authenticate or request for another form of payment

false	NO	SKIPPED_BY_BUYER	Buyer was presented the 3D Secure challenge but chose to skip the authentication	Do not continue with current authorization. Prompt the buyer to re-authenticate or request buyer for another form of payment
false	NO	FAILURE	Buyer may have failed the challenge or the device was not verified	Do not continue with current authorization. Prompt the buyer to re-authenticate or request buyer for another form of payment
false	NO	BYPASSED	3D Secure was skipped as authentication system did not require a challenge	You can continue with the authorization and assume liability. If you prefer not to assume liability, ask the buyer for another card
false	NO	ATTEMPTED	Card is not enrolled in 3D Secure. Card issuing bank is not participating in 3D Secure	Continue with authorization as authentication is not required
False	NO	UNAVAILABLE	Issuing bank is not able to complete authentication	You can continue with the authorization

				and assume liability. If you prefer not to assume liability, ask the buyer for another card
False	NO	CARD_INELIGIBLE	Card is not eligible for 3D Secure authentication	Continue with authorization as authentication is not required