# IATA BSP NDC API 3

# Business requirements & Technical Implementation Guide

# v1.1.2

| Document Version Control Information | |
|---|---|
| **Document No** | 1.1.2 |
| **Document Name** | IATA BSP NDC API 3 Business Requirements & Technical Implementation Guide |

| Version | Change Description | Sections | Date | Author | Reviewer |
|---|---|---|---|---|---|
| 1 | First version | All | 2023-08-24 | IATA Performance Operations | IATA Performance Operations |
| 1.1.1 | Reviewed | All | 2023-09-04 | IATA Performance Operations | IATA Performance Operations |
| 1.1.2 | ITS | All | 2023-09-08 | IATA Performance Operations | ITS |
| | | | | | |

# 1. NDC EASYPAY DIRECT AUTHORIZATION BUSINESS REQUIREMENTS

### 1.1 Background

With NDC, the airline is in control of the offer management, the order management and the NDC/One Order (OO) transaction issuance.

The airlines have outsourced to IATA the responsibility to manage the IATA Agency program to rely on a professional network of agents in more than 180 countries and territories.

They have also mandated IATA to manage the Billing and Settlement Plan (BSP) to facilitate and simplify the selling, reporting and remitting procedures of IATA Accredited Passenger Sales Agents.

IATA EasyPay is closed loop "pay as you go" e-wallet payment solution for agents to issue tickets via the BSP. To use IATA EasyPay, agents must fund their IATA EasyPay account. At the time of ticket issuance, the IATA EasyPay system will first verify that funds are available in the account. If this validation is successful, the system will generate an authorization and the related funds will be blocked in the agent's wallet. EasyPay is a BSP form of payment recognized in airline distribution standards and can be used for both GDS as well as NDC sales.

Please note that in BSP China, BSP Online Payment (BOP) is the EasyPay solution in that market.

Within this frame, one of the tools developed by IATA is an API that provides the BSP NDC participants with a direct authorization process for EasyPay.

With this API, NDC airlines could benefit from a simple and integrated process that facilitate EasyPay adoption in BSP NDC transactions, well integrated within the BSP and without the processing cost usually associated to payment authorizations for credit card payments.

### 1.2 NDC EasyPay Direct Authorization - Process description:

Prior to accepting IATA EasyPay form of payment details provided by a seller through the NDC messages, an authorization is required to make sure the seller funds will be the blocked on the seller's wallet for later settlement.

This API should be used by airlines to obtain such authorization.

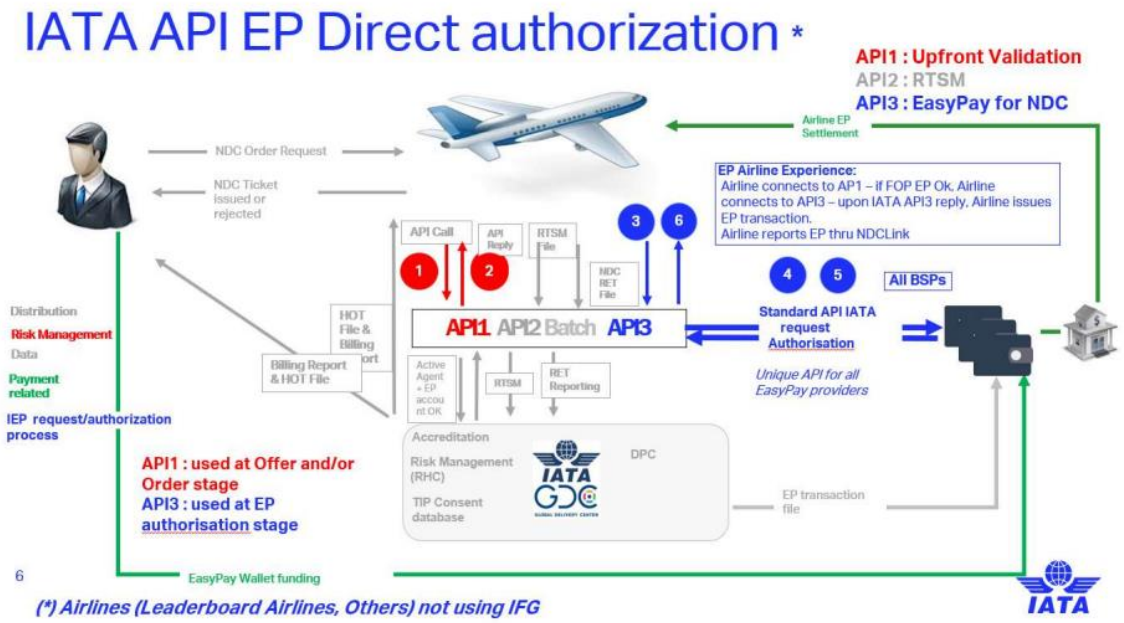
The EasyPay Direct Authorization (API3) consists of an exchange of data between three parties:

- NDC airline
- IATA Platform
- EasyPay provider.

Data real time exchanges are performed as following:

1. NDC Airline to IATA API platform
2. IATA API platform to the corresponding EasyPay provider (BSP specific)
3. EasyPay Provider responses back to IATA
4. IATA to response to NDC Airline.

The end goal of the solution is to provide to the NDC airline a means to validate the EasyPay payment details provided by the seller and secure the funds for later remittance.




In the flowchart above, we could see the IATA EasyPay Direct Authorization (AP3) calls, integrated in a complete NDC sale process, within sales reported to the BSP. The compulsory NDC Risk API calls (BSP Risk Monitoring Agency profile, API1 and BSP Risk Monitoring Agency Sales Transactions, API2) are also present in the graph.

The steps, 3, 4, 5 and 6 in the previous flowcharts, correspond to the API3 and represent the following actions:

- **Step 3**: NDC Airline -> IATA

  The airline communicates through API a set of minimum required data such as Agent identification, Airline identification, EasyPay amount requested for authorization and transfer from Agent EasyPay wallet, currency, transaction number. These data serve for the purpose of EasyPay authorization.

- **Step 4:** IATA -> EasyPay provider

  IATA transmits to the appropriate EP provider[1] the same information as in step 3.

- **Step 5:** EasyPay provider -> IATA

  EasyPay provider sends response for each authorization request which can be either "approved" or "declined". For approved authorization requests an approval code is generated and provided in the response.

- **Step 6:** IATA ->NDC Airline

  IATA communicates to NDC Airline info contained in step 5.

[1] Currently API3 is connected to the following EasyPay providers: IATA EasyPay 2.0, Trevipay and Edenred.

# 2. Technical Implementation Guide

## 2.1 General Information

IATA will provide two instances of the APIs. One for testing that we call sandbox, and the live production API. To support these two services, there will be two different URLs:

- Sandbox: https://apisbx.iata.org
- Production: https://api.iata.org

**Sandbox** instances of the APIs will require only **HTTP Basic Authorization** using a **client ID and client Secret** issued by IATA, to be provided as the standard HTTP Header "Authorization".

**Production** instances of the APIs will require only **HTTP Basic Authorization** using a **client ID and client Secret** issued by IATA, to be provided as the standard HTTP Header "Authorization".

Please note that sandbox and production credentials would be different.

Additionally, airline IP address should be whitelisted on IATA systems.

Currently IATA is working to adapt the API authentication process through Oauth2.0.

## 2.2 NDC EasyPay Direct Authorization API

The NDC EasyPay Direct Authorization API is only accessible to airlines participating in the BSP.

The web service is a RESTful API that allows NDC airlines, prior to accepting IATA EasyPay transaction, to interact with the corresponding EasyPay provider to block the seller required funds for later settlement. This API can be used by airlines to obtain instantly such authorization.

### Code and format accepted.

8-digit IATA numeric code / Travel Agency

### Technical Detail

URL: https://apisbx.iata.org/easypay-direct-authorisation/v1/agencies/{iataNumber}/transaction-authorisations

URI Parameters:

{iataNumber} = the IATA code issued to the agency

Query Parameters:

None required

Media/Mime-type:

application/json

HTTP Methods:

POST

**Please note that the examples are simulated/non-representative of real agents or tickets.**

Example Request Payload


```
{
"transactionAuthorisationType": "PAYMENT",
  "paymentId": "2019042913550400001",
  "orderId": "XB000ABCDEFG",
  "epNumber": "1234567890123456",
  "iataNumber": "98417900",
  "amount": 2900,
  "currencyCode": "USD",
  "requestTimestamp": "2019-04-29T13:55:31.378Z"
}
```


Example Response Payload (Authorized payment)


```
{
  "orderId": "XB000ABCDEFG",
  "paymentId": "2019042913550400001",
  "paymentStatus": "ACCEPTED",
  "approvalCode": "654321",
  "responseTimestamp": "2019-04-29T13:55:31.378Z"
}
```


Example Response Payload (Non-authorized payment: Wallet not found, or closed account)


```
{
  "id": "36f1169c-9f89-4c3c-916f-1efde37b70b1",
```

```
  "status": "400",
  "code": "NDA01",
  "title": "Account closed/no wallet found",
  "detail": "Payment authorization failed with Status F for orderId XB000ABCDEFG and
  paymentId 2019042913550400001 "
}
```


## Request Schema

requestBody:
 content:
  application/json:
   schema:
    $ref:  #/components/schemas/transactionAuthorisation

components:
  schemas:
   transactionAuthorisation:
    required:
- amount
- currencyCode
- epNumber
- iataNumber
- orderId
- paymentId
- requestTimestamp
- transactionAuthorisationType

  type: object,
  properties:

    transactionAuthorisationType:
     type: string,
     description: "Defines a type of authorization. Examples are: 'REFUND' or 'PAYMENT'. 'REFUND' is a reserved word, but currently only 'PAYMENT' has been developed",
     enum:
- REFUND
- PAYMENT

    paymentId:
     type: string,
     description : Uniquely identifies payment information within a message
     example:  2019042913550400001

    orderId:
     type: string,
     description : Carrier assigned ID which uniquely identifies a specific Order across several messages
     example:  XB000ABCDEFG

    epNumber:
     type: string,
     description: IATA Easy Pay Account Number
     maxLength : 16 digits,
    example: 1234567890123456

    iataNumber:

```
      pattern: ^[0-9]{8}$,
      type: string,
      description: IATA-assigned agency number.
      example: 98417900

    amount:
      minimum: 0,
      type: number,
      description:  The Transaction Amount Schema. It should be always a positive amount.
      example: 98417900

    currencyCode:
      maxLength : 3,
      minLength : 3,
      type: string,
      description: Currency codes (ISO 4217),
      example: USD

    requestTimestamp:
      type: string,
      description: The request timestamp Schema - Format rfc3339,
      format: date-time
      example: 2019-04-29T13:55:31.378Z
```

## Response Specification

```
responses:
  '200':
    description: OK
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/
    transactionAuthorisationResponse

  '400':
    description: Bad Request
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/errors

  '403':
    description:  Forbidden
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/errors


  '500':
    description: Internal Server Error
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/errors
```

```
responseBody:
 content:
  application/json:
   schema:
    $ref:  #/components/schemas/transactionAuthorisationResponse

components:
  schemas:
   transactionAuthorisationResponse:
    required:
     • approvalCode
     • orderId
     • paymentId
     • paymentStatus
     • responseTimestamp

  type: object,
  properties:

    paymentId:
     type: string,
     description: Uniquely identifies payment information within a message
     example: 2019042913550400001

    orderId:
     type: string,
     description: Carrier assigned ID which uniquely identifies a specific Order across several messages
     example:  XB000ABCDEFG

    paymentStatus:
     type: string,
     description: Indicates the current status of this payment information,
     enum:
       • ACCEPTED,
       • ALLOCATED,
       • CLOSED,
       • COMMITTED,
       • RECEIVED,
       •  REFUNDED,
       •  SENT


    approvalCode:
     type: string,
     description: Authorization result Approval code
     example: 654321

    responseTimestamp:
     type: string,
     description: The response timestamp Schema - Format rfc3339,
     format: date-time
     example: 2019-04-29T13:55:31.378Z


  errors:
   type: object
   properties:
```

```
    errors:
      type: array
      items:
        $ref: '#/components/schemas/error'

  error:
    required:
      status

    type: object
    properties:

      id:
        type: string
        description: A unique identifier for this specific instance of the error.

      status:
        type: string
        description: The HTTP status code applicable to the error.

      code:
        type: string
        description: An application-specific error code.

      title:
        type: string
        description: A short, human-readable summary of the problem that
      **SHOULD NOT** change from occurrence to occurrence of the error,
      except for purposes of localization short, human-readable summary of
      the problem.

      language:
        type: string
        description: The code of the language used in the error message. Not
      required when the language is a variant of English

      detail:
        type: string
        description: A human-readable explanation specific to this occurrence of the issue.

      url:
        type: string
        description: A link to an on-line description of the error where one COULD find statements pertaining to the
        consequences of the error and indications as to actions that might be taken and actions that should or must
        not be taken.
```

## Response Codes

As far as possible, any error response messages conform to the JSONAPI error object format. This is in accordance with the IATA Open Air Industry API working group's recommendations and certification check-list.

| HTTP Response Code | Remarks/Examples |
|---|---|
| 200 – OK | See Response payload described above |

| | |
|---|---|
| 400 – Bad request | {<br>"id": "36f1169c-9f89-4c3c-916f-1efde37b70b1",<br>"status": "400",<br>"code": "NDA01",<br>"title": "Account closed/no wallet found",<br>"detail": "Payment authorization failed with Status F for orderId XB000ABCDEFG and paymentId 2019042913550400001 "<br>}<br><br>Some other errors under 400:<br><br>• Wrong or malformed URI<br>• Malformed JSON object<br>• Payload does not comply with JSON schema<br>• IEP vendor of BSP operation of iataNumber is not integrated yet<br>• epNumber belongs to a closed wallet<br>• iataNumber does not have an IEP Status "Open" in AMS<br>• epNumber belongs to a wallet of a closed account<br>• epNumber not generated or recognized by IEP vendor<br>• epNumber belongs to a wallet not owned by iataNumber<br>• airlineId is not a known ACLI airline code<br>• The number of decimals in the amount field exceed those of the currency as defined in the ISO 4217 standard.<br>• CurrencyCode does not match the currency of the wallet<br>• The amount requested is higher than the available funds on the wallet<br>• Combination of orderId and paymentId have been used in prior successful authorisation |
| 401 - Unauthorized | Some errors under 401:<br><br>• No credentials provided<br>• Invalid credentials |
| 403 – Forbiden | {<br>"id": "36f1169c-9f89-4c3c-916f-1efde37b70b1",<br>"status": "403",<br>"code": "Forbidden",<br>"title": "Forbidden",<br>}<br><br>Other errors under 403:<br>• Credentials lack sufficient permissions |

| | |
|---|---|
| 500 – Server error | An indeterminate problem occurred in the back-end data processing systems. There may or may not be additional detail in the response payload. This condition is monitored by the API infrastructure but should also be reported to IATA if it occurs.<br>{<br>"id": "36f1169c-9f89-4c3c-916f-1efde37b70b1",<br>"status": "500",<br>"code": "InternalServerError",<br>"title": "Internal server Error",<br>"detail": "Failed to process payment. Error: unknown exception"<br>} |