API REFERENCE DOCUMENTATION

Go2Solve FIRS Integrators API

RESTful integration guide for third-party partners connecting to the Federal Inland Revenue Service (FIRS) e-Invoice system

Base URL (Production): api.go2solve.com/firs/integrators
Base URL (Staging): staging-api.go2solve.com/firs/integrators

Table of Contents

Go2Solve FIRS Integrators API Reference

Overview

The **Go2Solve FIRS Integrators API** provides a RESTful interface for third-party integrators to interact with the Federal Inland Revenue Service (FIRS) e-Invoice system. This API enables partners to:

- **Validate and confirm Invoice Reference Numbers (IRN)**
- **Submit and sign invoices** for FIRS compliance
- **Update invoice payment status**
- **Generate QR codes** for invoices
- **Download invoice documents**
- **Access reference data** (invoice types, product codes, tax categories, etc.)

Base URL

```
Production: https://api.go2solve.com/firs/integrators
Staging:    https://staging-api.go2solve.com/firs/integrators
```

Authentication

All API requests require authentication via an **API Key** passed in the request header.

API KEY AUTHENTICATION

Header Name	Description
`X-API-KEY`	Your unique API key provided during onboarding

EXAMPLE REQUEST HEADER

```
GET /api/resources/invoice-types HTTP/1.1
Host: api.go2solve.com
X-API-KEY: your-api-key-here
Content-Type: application/json
```

AUTHENTICATION ERRORS

Status Code	Message	Description
`401`	API Key was not provided	Missing `X-API-KEY` header
`401`	Invalid API Key	Empty or malformed API key
`401`	Unauthorized client	API key not found or inactive

Response Format

All API responses follow a standard envelope format.

SUCCESS RESPONSE

```
{
  "statusCode": 200,
  "succeeded": true,
  "message": "Operation completed successfully",
  "errors": [],
  "data": { }
}
```

ERROR RESPONSE

```json
{
  "statusCode": 400,
  "succeeded": false,
  "message": "Validation failed",
  "errors": [
    "Field 'irn' is required",
    "Invalid invoice type code"
  ],
  "data": null
}
```

Endpoints

Invoice Operations

1. Confirm IRN

Validates and confirms an Invoice Reference Number with the FIRS system.

POST `/api/invoices/confirm-irn`

Authentication: Required

REQUEST BODY

Field	Type	Required	Description
`invoiceReference`	`string`	**Yes**	Your internal invoice reference
`irn`	`string`	**Yes**	Invoice Reference Number to validate

REQUEST EXAMPLE

```
curl -X POST "https://api.go2solve.com/firs/integrators/api/invoices/confirm-irn" \
  -H "X-API-KEY: your-api-key" \
  -H "Content-Type: application/json" \
  -d '{
    "invoiceReference": "INV-2024-001",
    "irn": "FIRS-IRN-123456789"
  }'
```

RESPONSE EXAMPLE (SUCCESS)

```
{
  "statusCode": 200,
  "succeeded": true,
  "message": "IRN confirmed successfully",
  "errors": [],
  "data": { "ok": true }
}
```

RESPONSE EXAMPLE (ERROR)

```
{
  "statusCode": 400,
  "succeeded": false,
  "message": "IRN validation failed",
  "errors": ["IRN not found in FIRS system"],
  "data": null
}
```

2. Confirm Invoice

Submits an invoice for confirmation with the FIRS system.

POST `/api/invoices/confirm-invoice`

Authentication: Required

REQUEST BODY

Field	Type	Required	Description
`irn`	`string`	**Yes**	Invoice Reference Number
`issueDate`	`string` (date)	**Yes**	Invoice issue date (YYYY-MM-DD)
`dueDate`	`string` (date)	No	Payment due date

`issueTime`	`string` (time)	No	Invoice issue time (HH:mm:ss)
`invoiceTypeCode`	`string`	No	Type of invoice (see Resources)
`paymentStatus`	`string`	No	Current payment status
`note`	`string`	No	Additional notes
`documentCurrencyCode`	`string`	No	Currency code (e.g., "NGN")
`taxCurrencyCode`	`string`	No	Tax currency code
`accountingSupplierParty`	`object`	No	Supplier details
`accountingCustomerParty`	`object`	No	Customer details
`taxTotal`	`array`	No	Tax breakdown
`legalMonetaryTotal`	`object`	No	Invoice totals
`invoiceLine`	`array`	No	Line items

SUPPLIER / CUSTOMER PARTY OBJECT

Field	Type	Description
`partyName`	`string`	Business name
`tin`	`string`	Tax Identification Number
`email`	`string`	Contact email
`telephone`	`string`	Phone number
`businessDescription`	`string`	Business description
`postalAddress`	`object`	Address details

POSTAL ADDRESS OBJECT

Field	Type	Description
`streetName`	`string`	Street address
`cityName`	`string`	City
`postalZone`	`string`	Postal / ZIP code
`country`	`string`	Country code

REQUEST EXAMPLE

```
curl -X POST "https://api.go2solve.com/firs/integrators/api/invoices/confirm-invoice" \
  -H "X-API-KEY: your-api-key" \
  -H "Content-Type: application/json" \
  -d '{
    "irn": "FIRS-IRN-123456789",
    "issueDate": "2024-01-15",
    "dueDate": "2024-02-15",
    "invoiceTypeCode": "380",
    "paymentStatus": "PENDING",
    "documentCurrencyCode": "NGN",
    "accountingSupplierParty": {
      "partyName": "Acme Corporation Ltd",
      "tin": "12345678-0001",
      "email": "billing@acme.com",
      "telephone": "+234-1-234-5678",
      "businessDescription": "Software Development",
      "postalAddress": {
        "streetName": "123 Victoria Island",
        "cityName": "Lagos",
        "postalZone": "101001",
        "country": "NG"
      }
    },
    "accountingCustomerParty": {
      "partyName": "Customer Company Ltd",
      "tin": "87654321-0001",
      "email": "accounts@customer.com",
```

```
      "telephone": "+234-1-876-5432",
      "businessDescription": "Retail",
      "postalAddress": {
        "streetName": "456 Ikeja GRA",
        "cityName": "Lagos",
        "postalZone": "100001",
        "country": "NG"
      }
    },
    "invoiceLine": [
      {
        "lineId": "1",
        "quantity": 10,
        "unitCode": "EA",
        "lineExtensionAmount": 50000.00,
        "itemName": "Software License",
        "price": 5000.00
      }
    ],
    "legalMonetaryTotal": {
      "lineExtensionAmount": 50000.00,
      "taxExclusiveAmount": 50000.00,
      "taxInclusiveAmount": 57500.00,
      "payableAmount": 57500.00
    }
  }'
```

RESPONSE EXAMPLE (SUCCESS)

```
{
  "statusCode": 200,
  "succeeded": true,
  "message": "Invoice confirmed successfully",
  "errors": [],
  "data": { "ok": true }
}
```

3. Sign Invoice

Digitally signs an invoice for FIRS submission.

POST `/api/invoices/sign-invoice`

Authentication: Required

REQUEST BODY

Same as **Confirm Invoice** (see endpoint 2 above).

REQUEST EXAMPLE

```
curl -X POST "https://api.go2solve.com/firs/integrators/api/invoices/sign-invoice" \
  -H "X-API-KEY: your-api-key" \
  -H "Content-Type: application/json" \
  -d '{
    "irn": "FIRS-IRN-123456789",
    "issueDate": "2024-01-15",
    "invoiceTypeCode": "380",
    "documentCurrencyCode": "NGN"
  }'
```

RESPONSE EXAMPLE

```
{
  "statusCode": 200,
  "succeeded": true,
  "message": "Invoice signed successfully",
  "errors": [],
  "data": { "ok": true }
}
```

4. Update Invoice

Updates the payment status of an existing invoice.

PATCH `/api/invoices/update-invoice/{irn}`

Authentication: Required

PATH PARAMETERS

Parameter	Type	Required	Description
`irn`	`string`	**Yes**	Invoice Reference Number

REQUEST BODY

Field	Type	Required	Description
`reference`	`string`	No	Payment reference
`PaymentStatus`	`enum`	**Yes**	New payment status

PAYMENT STATUS VALUES

Value	Description
`PENDING`	Payment not yet received
`PAID`	Payment completed
`CANCELLED`	Invoice cancelled

REQUEST EXAMPLE

```
curl -X PATCH "https://api.go2solve.com/firs/integrators/api/invoices/update-invoice/FIRS-IRN-123456789" \
  -H "X-API-KEY: your-api-key" \
  -H "Content-Type: application/json" \
```

```
  -d '{
    "reference": "PAY-REF-001",
    "PaymentStatus": "PAID"
  }'
```

RESPONSE EXAMPLE


```
{
  "statusCode": 200,
  "succeeded": true,
  "message": "Invoice updated successfully",
  "errors": [],
  "data": { "ok": true }
}
```


5. Generate QR Code

Generates a QR code for an invoice.

GET `/api/invoices/generate-qr-code/{irn}`

Authentication: Required

PATH PARAMETERS

Parameter	Type	Required	Description
`irn`	`string`	**Yes**	Invoice Reference Number

REQUEST EXAMPLE

```
curl -X GET "https://api.go2solve.com/firs/integrators/api/invoices/generate-qr-code/FIRS-IRN-123456789" \
  -H "X-API-KEY: your-api-key"
```

RESPONSE EXAMPLE


```
{
  "statusCode": 200,
  "succeeded": true,
  "message": "QR code generated",
  "errors": [],
  "data": "data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAA..."
}
```


6. Download Invoice

Downloads the invoice document as a file.

GET `/api/invoices/download-invoice/{irn}`

Authentication: Required

PATH PARAMETERS

Parameter	Type	Required	Description
`irn`	`string`	**Yes**	Invoice Reference Number

REQUEST EXAMPLE

```
curl -X GET "https://api.go2solve.com/firs/integrators/api/invoices/download-invoice/FIRS-IRN-123456789" \
  -H "X-API-KEY: your-api-key" \
  --output invoice.pdf
```

RESPONSE

Returns the invoice file directly with the appropriate content type (e.g., `application/pdf`).

Resource Endpoints

These endpoints provide reference data required for invoice creation.

7. Get Invoice Types

Returns the list of valid invoice type codes.

GET /api/resources/invoice-types

Authentication: Required

REQUEST EXAMPLE

```
curl -X GET "https://api.go2solve.com/firs/integrators/api/resources/invoice-types" \
  -H "X-API-KEY: your-api-key"
```

RESPONSE EXAMPLE

```
{
  "statusCode": 200,
  "succeeded": true,
  "message": "",
  "errors": [],
  "data": [
    { "code": "380", "value": "Commercial Invoice" },
    { "code": "381", "value": "Credit Note" },
    { "code": "383", "value": "Debit Note" },
    { "code": "386", "value": "Prepayment Invoice" }
  ]
}
```

8. Get Product Codes

Returns the list of HS (Harmonized System) product codes.

GET /api/resources/product-codes

Authentication: Required

REQUEST EXAMPLE

```
curl -X GET "https://api.go2solve.com/firs/integrators/api/resources/product-codes" \
  -H "X-API-KEY: your-api-key"
```

RESPONSE EXAMPLE

```
{
  "statusCode": 200,
  "succeeded": true,
  "message": "",
  "errors": [],
  "data": [
    { "hscode": "0101", "description": "Live horses, asses, mules and hinnies" },
    { "hscode": "8471", "description": "Automatic data processing machines" }
  ]
}
```

9. Get Service Codes

Returns the list of service classification codes.

GET /api/resources/service-codes

Authentication: Required

REQUEST EXAMPLE

```
curl -X GET "https://api.go2solve.com/firs/integrators/api/resources/service-codes" \
  -H "X-API-KEY: your-api-key"
```

RESPONSE EXAMPLE


```
{
  "statusCode": 200,
  "succeeded": true,
  "message": "",
  "errors": [],
  "data": [
    { "code": "9701", "description": "Software Development Services" },
    { "code": "9702", "description": "IT Consulting Services" }
  ]
}
```

10. Get Tax Categories

Returns the list of tax category codes.

GET /api/resources/tax-categories

Authentication: Required

REQUEST EXAMPLE

```
curl -X GET "https://api.go2solve.com/firs/integrators/api/resources/tax-categories" \
  -H "X-API-KEY: your-api-key"
```

RESPONSE EXAMPLE


```
{
  "statusCode": 200,
  "succeeded": true,
  "message": "",
  "errors": [],
  "data": [
    { "code": "S", "value": "Standard Rate" },
    { "code": "Z", "value": "Zero Rated" },
    { "code": "E", "value": "Exempt" }
  ]
}
```


11. Get VAT Exemptions

Returns the list of VAT exemption categories.

GET /api/resources/vat-exemptions

Authentication: Required

REQUEST EXAMPLE

```
curl -X GET "https://api.go2solve.com/firs/integrators/api/resources/vat-exemptions" \
  -H "X-API-KEY: your-api-key"
```

RESPONSE EXAMPLE


```
{
  "statusCode": 200,
  "succeeded": true,
  "message": "",
  "errors": [],
  "data": [
    {
      "headingNo": "01",
      "harmonizedSystemCode": "0101.21",
      "tariffCategory": "A",
      "tariff": "0%",
      "description": "Basic food items - Exempt"
    }
  ]
}
```


12. Get Currencies

Returns the list of supported currencies.

GET /api/resources/currencies

Authentication: Required

REQUEST EXAMPLE

```
curl -X GET "https://api.go2solve.com/firs/integrators/api/resources/currencies" \
  -H "X-API-KEY: your-api-key"
```

RESPONSE EXAMPLE


```
{
  "statusCode": 200,
  "succeeded": true,
  "message": "",
  "errors": [],
  "data": [
    {
      "symbol": "\u20a6",
      "name": "Nigerian Naira",
      "symbolNative": "\u20a6",
      "decimalDigits": 2,
      "rounding": 0,
      "code": "NGN",
      "namePlural": "Nigerian nairas"
    },
    {
      "symbol": "$",
      "name": "US Dollar",
      "symbolNative": "$",
      "decimalDigits": 2,
      "rounding": 0,
      "code": "USD",
      "namePlural": "US dollars"
    }
  ]
}
```

13. Get Local Governments

Returns the list of Nigerian local government areas.

GET /api/resources/local-governments

Authentication: Required

REQUEST EXAMPLE

```
curl -X GET "https://api.go2solve.com/firs/integrators/api/resources/local-governments" \
  -H "X-API-KEY: your-api-key"
```

RESPONSE EXAMPLE


```
{
  "statusCode": 200,
  "succeeded": true,
  "message": "",
  "errors": [],
  "data": [
    { "name": "Ikeja", "code": "LG001", "stateCode": "LA" },
    { "name": "Victoria Island", "code": "LG002", "stateCode": "LA" }
  ]
}
```


14. Get States

Returns the list of Nigerian states.

GET /api/resources/states

Authentication: Required

REQUEST EXAMPLE

```
curl -X GET "https://api.go2solve.com/firs/integrators/api/resources/states" \
  -H "X-API-KEY: your-api-key"
```

RESPONSE EXAMPLE


```
{
  "statusCode": 200,
  "succeeded": true,
  "message": "",
  "errors": [],
  "data": [
    { "name": "Lagos", "code": "LA" },
    { "name": "Abuja FCT", "code": "FC" },
    { "name": "Rivers", "code": "RI" }
  ]
}
```


15. Get Payment Means

Returns the list of valid payment methods.

GET /api/resources/payment-means

Authentication: Required

REQUEST EXAMPLE

```
curl -X GET "https://api.go2solve.com/firs/integrators/api/resources/payment-means" \
  -H "X-API-KEY: your-api-key"
```

RESPONSE EXAMPLE


```json
{
  "statusCode": 200,
  "succeeded": true,
  "message": "",
  "errors": [],
  "data": [
    { "code": "10", "value": "Cash" },
    { "code": "30", "value": "Credit Transfer" },
    { "code": "42", "value": "Bank Card" },
    { "code": "48", "value": "Bank Account" }
  ]
}
```

Error Codes

HTTP Status Codes

Status Code	Description
`200`	Success — Request completed successfully
`400`	Bad Request — Invalid request body or parameters
`401`	Unauthorized — Missing or invalid API key
`404`	Not Found — Resource not found
`422`	Unprocessable Entity — Validation error
`429`	Too Many Requests — Rate limit exceeded
`500`	Internal Server Error — Server-side error

Common Error Messages

Error Message	Cause	Solution
`API Key was not provided`	Missing X-API-KEY header	Add the X-API-KEY header to your request
`Invalid API Key`	Empty API key value	Provide a valid API key
`Unauthorized client`	API key not found or inactive	Contact support to verify your API key status
`IRN not found`	Invoice reference doesn't exist	Verify the IRN is correct
`Invoice already confirmed`	Duplicate confirmation attempt	Invoice has already been processed
`Validation failed`	Request body validation error	Check the `errors` array for specific field errors

Best Practices

Security

1. **Protect your API Key**: Store your API key securely (e.g., environment variables, secrets manager). Never expose it in client-side code.
2. **Use HTTPS**: Always use HTTPS endpoints to encrypt data in transit.
3. **Validate responses**: Always check the `succeeded` field and handle errors appropriately.

Performance

1. **Cache reference data**: Resources like invoice types, currencies, and states change infrequently. Cache these responses to reduce API calls.
2. **Respect rate limits**: Implement exponential backoff when you receive `429` responses.
3. **Batch operations**: When processing multiple invoices, consider implementing a queue system to stay within rate limits.

Reliability

1. **Implement retry logic**: For network failures or 5xx errors, implement retry with exponential backoff.
2. **Log API interactions**: Maintain logs of API requests and responses for debugging and audit purposes.
3. **Handle timeouts**: Set appropriate timeout values (recommended: 30 seconds) and handle timeout exceptions gracefully.

Integration Tips

1. **Test in staging first**: Use the staging environment to test your integration before going to production.

2. **Validate data before submission**: Validate invoice data locally before sending to the API to reduce failed requests.
3. **Use reference endpoints**: Always use the reference endpoints (invoice types, product codes, etc.) to ensure you're using valid codes.
4. **Monitor API health**: Subscribe to status updates and monitor the health endpoint for service availability.

Support

For technical support or API access requests:

Email	api-support@go2solve.com
Documentation	https://docs.go2solve.com/firs
Status Page	https://status.go2solve.com