Send Invoice

Introduction

This API acts as the bridge between your application and **Front Payment**, making it effortless to create invoices automatically. Instead of manually entering customer and order information, you can simply send the details directly from your system. This ensures a smooth and quick process, so that once a transaction or order is complete, a precise invoice is instantly generated and delivered to the right customer. It streamlines your billing process and helps you keep everything organized.

The **invoice distribution method** is determined based on the information provided in the request. The system follows this priority order:

1. **EHF or E-invoice (Preferred)** – Requires a valid P-number or organization number.
2. **Email** – Used if EHF/E-invoice cannot be delivered.
3. **Postal Mail** – Used if no valid email address is available or email delivery fails.

If none of the above delivery methods are successful, our customer service team will notify the client to resolve the issue.

Use Cases

Here are a few ways your application can use this API:

- **Automated E-commerce Billing**
  When a customer completes a purchase on your online store, your application can instantly use this API to generate and send an invoice to their email, creating a seamless and immediate billing process.
- **Subscription Management**
  For services with recurring billing, the API can be used to automatically generate and send invoices to customers at regular intervals (e.g., monthly or annually), eliminating the need for manual billing cycles.
- **Service & Project Invoicing**
  After a project or service is completed, your application can use the API to automatically draft and send a detailed invoice to the client, including all project details, labor costs, and materials.
- **Financial & Accounting System Integration**
  Your application can use this API to sync order and billing data directly with an accounting system, ensuring all financial records are up-to-date and accurate without manual data

entry.

Prerequisites

Before you start the integration, make sure you have:

1. API Access:

- A valid API key and Bearer Token from Front Payment
- Access to the demo and production environments

2. Technical Requirements:

- Ability to make HTTPS API calls
- Secure storage of tokens and keys
- Callback endpoints to handle payment status updates

3. Test Environment:

- For testing, contact `nafees.faraz@frontpayment.no` to gain access to the demo environment

Endpoint

```
POST https://demo-api.frontpayment.no/api/v1/connect/orders/invoice/create
```

Authorization

Include a **Bearer Token** in the `Authorization` header. You can obtain this token from **Front Payment**.

Example:

```
Authorization: Bearer YOUR_FRONTPAYMENT_BEARER_TOKEN
```

Request Payload

Send the following parameters as a JSON object in the request body:

```
{
  "products": [
    {
```

```
        "name": "Hair Wash",
        "productId": "VFDDF",
        "quantity": "1",
        "rate": 51,
        "discount": 0,
        "tax": "0",
        "amount": 51
    }
],
"orderSummary": {
    "subTotal": "51.00",
    "totalTax": "0.00",
    "totalDiscount": "0.00",
    "grandTotal": "51.00"
},
"orderDate": "1703040812",
"customerDetails": {
    "type": "private",
    "countryCode": "+47",
    "msisdn": "46567468",
    "email": "test@yopmail.com",
    "firstName": "",
    "name": "Kari Nordmann",
    "preferredLanguage": "en",
    "personalNumber": "12345678901,
    "organizationId": null,
    "address": {
        "street": "Luramyrveien 65",
        "zip": "4313",
        "city": "Sandnes",
        "country": "NO"
    }
},
"invoiceInterval": 0,
"invoiceMaturity": 10,
"invoiceFeeApplicable": true,
"separateInvoices": true,
"referenceNo": null,
"customerReference": null,
```

```
    "callback": {
        "callbackUrl": "https://example.com/callback-url"
    }
}
```

Validation Rules

Make sure your request meets the following requirements:

Field	Type	Description
products.*.name	string	**Required** Name of the product.
products.*.productId	string	**Optional** Unique identifier for the product.
products.*.quantity	numeric	**Required** Quantity of the product.
products.*rate	numeric	**Required** Rate per unit of the product.
products.*.discount	numeric	**Optional** Discount applied to the product.
products.*.tax	numeric	**Required** Tax rate must be (e.g., `0`, `12`, `15`, `25`), Unless you have other configuration.
products.*.amount	numeric	**Required** Total amount for the product line item.
orderSummary.subTotal	numeric	**Required** Subtotal of all products before tax and discount.
orderSummary.totalTax	numeric	**Required** Total tax for the order.
orderSummary.totalDiscount	numeric	**Required** Total discount for the order.
orderSummary.grandTotal	numeric	**Required** Grand total of the order.
orderDate	string	**Required** Unix timestamp for the Date of the order.
customerDetails.countryCode	string	**Required** Country code for the customer's phone number (e.g., "+47").
customerDetails.msisdn	string	**Required** Mobile Subscriber ISDN Number (phone number).
customerDetails.email	email	**Required** Customer's email address.
customerDetails.firstName	email	**Optional** Customer's first name.
customerDetails.name	email	**Required** Customer's full name.

Field	Type	Description
`customerDetails.preferredLanguage`	`string`	**Optional** Customer preferred language. Available languages are `en`, `no`, `sv`, `da`, `de`. If nothing is given it will set default to `no`.
`customerDetails.type`	`string`	**Required** You can provide customer type is `private` or `corporate`.
`customerDetails.personalNumber`	`string`	**Optional** Customer's personal identification number, must be exactly 11 characters containing only numbers and cannot contain spaces. When Customer type is `private` then you can used this for add personal number.
`customerDetails.organizationId`	`string`	**Conditional Required** Organization identification number, must contain only numbers and cannot contain spaces. When Customer type is `corporate` then this field is required. Otherwise you can add this as `null` or remove from payload.
`customerDetails.address.street`	`string`	**Required** Street address of the customer.
`customerDetails.address.zip`	`string`	**Required** Zip code of the customer's address.
`customerDetails.address.city`	`string`	**Required** City of the customer's address.
`customerDetails.address.country`	`string`	**Required** ISO Alpha-2 country code (e.g., `NO`). Custom validation `IsoAlpha2Country` applies.
`invoiceInterval`	`numeric`	**Optional** Default value is = `0`. You can change it to `0`, `1`, `2`. Daily = `0`, Once a month = `1`, Twice a month = `2`.
`invoiceFeeApplicable`	`boolean`	**Required** Default value is `true`.
`invoiceMaturity`	`numeric`	**Optional** If specified, the value must be `10` for private customers; for corporate customers, the value may be `14`, `30`, or `45`.
`separateInvoices`	`boolean`	**Optional** Default value is `true`.
`referenceNo`	`string`	**Nullable** Any reference number.
`customerReference`	`string`	**Nullable** Any value for customer reference.

Field	Type	Description
`callback.callbackUrl`	`url`	**Optional** To receive real-time notifications on order state changes, you must provide a callback url. This is an server-to-server `HTTP GET` request.

Response

A successful request will return a `201 Created` status with the following JSON payload:


```
{
   "status_code": 201,
   "status_message": "OK",
   "message": "orderAddedSuccessfully",
   "is_data": false,
   "data": {
      "uuid": "ODR3506777330",
      "customerUuid": "CSRT3463048878"
   }
}
```


API returns a `500` or `510` error, it means something failed on the server side


```
{
   "status_code": 500,
   "status_message": "Internal Dependency Error",
   "message": "internalErrorOccurredPleaseTryAgainLater",
   "is_error": true,
   "errors": {
      "happenedAt": "String",
      "internalErrorDetails": "Array"
   }
}
```

```
{
   "status_code": 510,
   "status_message": "Execution Exception Occurred",
   "message": "somethingWentWrong",
   "is_error": true,
```

```
    "errors": "Array"
}
```


Notifications via Callback URL

The `callbackUrl` is an endpoint on your server that our system will call via an `HTTP GET` request whenever the status of the specified order changes from its initial state.

See the link below to understand how to work with the callback URL on your side and how to verify the request sent from our side.

Go To `Notication Via Callback Url` Page

Revision #50
Created 8 January 2024 17:14:54 by Admin
Updated 19 March 2026 09:57:01 by Admin