Create Checkout Session - Card, Vipps, Apple & Google Pay

The Create Checkout Session API enables merchants to generate secure, one-time checkout sessions for customers. This ensures a quick and PCI-compliant payment process without requiring the customer to create an account or save a payment method.

Key use cases

- **E-commerce**: Generate links for one-off product sales.
- **Services**: Request upfront payments (consulting, events, classes).

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. Merchant Setup:

- Your merchant account configured with Front Payment
- Enabled payment methods (Vipps, Google Pay, Apple Pay, Visa, Mastercard)

3. Technical Requirements:

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

4. Test Environment:

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

Step 1: Create Payment URL

To initiate a payment, your system will need to call our create endpoint to generate a payment URL. This URL will redirect your users to payment gateway.

Endpoint

```
POST https://demo-api.frontpayment.no/api/v1/connect/orders/regular/submit
```

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

The request body should be a JSON object containing details about the order, customer, and callback URLs.


```
{
 "products": [
  {
   "name": "Test Product",
   "productId": "1234",
   "quantity": 1,
   "rate": 4500,
   "discount": 0,
   "tax": 12,
   "amount": 4500
  }
 ],
 "orderSummary": {
  "subTotal": 4017.86,
  "totalTax": 482.14,
  "totalDiscount": 0.00,
  "grandTotal": 4500.00,
```

```
    "shippingCost": 0.00
  },
  "referenceNo": "",
  "customerReference": "",
  "orderDate": "1754556624",
  "withCustomer": true,
  "customerDetails": {
    "type": "private",
    "countryCode": "+47",
    "msisdn": "46567468",
    "email": "kari@nordmann.no",
    "name": "Kari Nordmann",
    "preferredLanguage": "en",
    "personalNumber": null,
    "organizationId": null,
    "address": {
      "street": "Luramyrveien 65",
      "zip": "4313",
      "city": "Sandnes",
      "country": "NO"
    }
  },
  "submitPayment": {
    "via": "visa"
  },
  "callback": {
    "callbackUrl": "https://your-callback-url.com/callback",
    "success": "https://your-callback-url.com/success",
    "failure": "https://your-callback-url.com/failure"
  }
}
```


Validation Rules

Ensure your payload adheres to the following validation rules:

Field	Type	Description
products.*.name	string	**Required**. The name of the product.

Field	Type	Description
`products.*.productId`	`string`	**Optional**. The unique ID of 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 unless otherwise configured.
`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**. The total tax for the order.
`orderSummary.totalDiscount`	`numeric`	**Required**. Total discount for the order.
`orderSummary.grandTotal`	`numeric`	**Required**. Grand total of the order.
`orderSummary.shippingCost`	`numeric`	**Optional**. Shipping cost of order.
`orderDate`	`string`	**Required**. Unix timestamp for the Date of the order, which must be current or future date.
`referenceNo`	`string`	**Optional**. Any reference information from your side. example: Order Uuid generated from your application.
`customerReference`	`string`	**Optional**. Customer reference
`orderFrom`	`string`	**Conditionally Required** if `fpgoUuid` is present. If provided, the value must be `PARTNER`. This indicates that the request originates from a registered partner and is intended to update an existing record.
`fpgoUuid`	`string`	**Optional** Use this to **prevent duplicates**. Pass the `orderUuid` from a previous response to update that specific order. If omitted, a new order is created.
`withCustomer`	`boolean`	**Required**. If **withCustomer** is true then you must provide customer details

Field	Type	Description
`customerDetails.type`	`string`	The customer type. **Required** if **withCustomer** is *true*. Must be either `private` or `corporate`.
`customerDetails.countryCode`	`string`	Country code for the customer's phone number (e.g., "+47"). **Required** if **withCustomer** is *true*.
`customerDetails.msisdn`	`string`	Mobile Subscriber MSISDN Number (phone number). **Required** if **withCustomer** is *true*.
`customerDetails.email`	`string`	Customer's email address. **Required** if **withCustomer** is *true*.
`customerDetails.name`	`string`	Customer's full name. **Required** if **withCustomer** is *true*.
`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.personalNumber`	`string`	**Optional**. Customer's personal identification number, must be 11 characters.
`customerDetails.organizationId`	`numeric`	**Required** if customer type is *corporate*. Must be alphanumeric.
`customerDetails.address.street`	`string`	Street address of the customer. **Required** if **withCustomer** is *true*.
`customerDetails.address.zip`	`string`	Zip code of the customer's address. **Required** if **withCustomer** is *true*.
`customerDetails.address.city`	`string`	City of the customer's address. **Required** if **withCustomer** is *true*.
`customerDetails.address.country`	`string`	ISO Alpha-2 country code (e.g., "NO"). Custom validation IsoAlpha2Country applies. **Required** if **withCustomer** is *true*.
`submitPayment.via`	`string`	**Required**. The payment method. Available payment methods `vipps`, `visa`, `mastercard`, `applepay`, or `googlepay`.
`callback.callbackUrl`	`url`	**Required**. The URL to which Front Payment will send updates. Must be a valid url.
`callback.success`	`url`	**Required**. The URL to redirect to upon successful payment. Must be a valid url.

Field	Type	Description
`callback.failure`	`url`	**Required**. The URL to redirect to upon failed payment. Must be a valid url.

Response

Success Response (HTTP 201)

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


```
{
  "status_code": 201,
  "status_message": "OK",
  "message": "Order Submitted Successfully",
  "is_data": true,
  "data": {
    "orderUuid": "ODR123456789",
    "customerUuid": "CSRT40567996",
    "paymentUrl": "https://v1.checkout.bambora.com/a403d3df20af4888bd8f7dd38f3cd7f1"
  }
}
```


Error Responses

HTTP 500: Internal Dependency Error


```
{
  "status_code": 500,
  "status_message": "Internal Dependency Error",
  "message": "Internal Error Occurred Please Try Again Later",
  "is_error": true,
  "errors": {
    "happenedAt": "String",
    "internalErrorDetails": "Array"
  }
}
```


HTTP 510: Execution Exception

```
{
  "status_code": 510,
  "status_message": "Execution Exception Occurred",
  "message": "Something Went Wrong",
  "is_error": true,
  "errors": "Array"
}
```


Step 2: Redirect to the Payment Gateway

After you successfully complete Step 1, you'll receive a **paymentUrl**. Redirect the user to this payment gateway, so they can make payment and complete the transaction.

After the user completes their payment, our system redirects them back to your application:

- If the payment is successful, they are redirected to the **success URL** you provided.
- If the payment fails, they are redirected to the **failure URL** you provided.

Additionally, our system will send a notification to the **callbackUrl** you gave in your initial request payload, updating your system on the payment status.

Notifications via Callback URL

For `paymentLink` order, after payment completed successfully, we will notify your server via the `callbackUrl` provided by you. Follow the link below to learn how to handle callback data from your side.

Go To `Notication Via Callback Url` Page

Best Practices

- Always validate amounts on your backend before marking payment as successful.
- Use **webhooks (callbackUrl)** as your source of truth, not just redirects.
- Ensure `orderDate` is a valid Unix timestamp and not expired.
- For corporate customers, `organizationId` is mandatory.

Revision #91
Created 7 August 2023 14:36:20 by Admin
Updated 14 April 2026 05:58:11 by Admin