Job API - Get and Insert Option

Product Configuration, Pricing & Order Submission

1. Overview

The BlindMatrix Ecommerce API enables partners to:

- Retrieve ecommerce-enabled products
- Dynamically configure products using field schemas
- Load selectable options and dependent subfields
- Validate dimensions (width/drop)
- Apply business rules and pricing formulas
- Calculate VAT and final pricing
- Submit completed orders with structured configuration data (`jsondata`)

This API is designed to support **complex configurable products** such as blinds, fabrics, and accessories.

2. Authentication & Headers

All API requests require the following headers unless explicitly stated otherwise.

Common Headers

```
{
  "Accept": "application/json",
  "Content-Type": "application/json",
  "companyname": "{api_name}",
  "Ecommercekey": "{api_key}",
```

```
  "platform": "Ecommerce"
}
```

Credentials

Field	Description
companyname	Provided by BlindMatrix support
Ecommercekey	Secret API key provided by BlindMatrix support

3. Get API URL Pointer

Endpoint

`POST https://blindmatrix.software/companydomains/getapidomain.php`

Description

Retrieves the base API URLs required for all subsequent requests.

Request Body

```
{
  "companyname": "CompanyDB"
}
```

Response

```
{
  "apipointer": "https://api.example.com/api",
  "nodeapipointer": "https://api.example.com/nodeapi"
}
```

Field	Description

apipointer	Base URL for public APIs
nodeapipointer	Base URL for Node APIs

4. Product Discovery

4.1 Get Product Details

Endpoint

```
GET {apipointer}/public/api/getproductsdetails/{productid}
```

Description

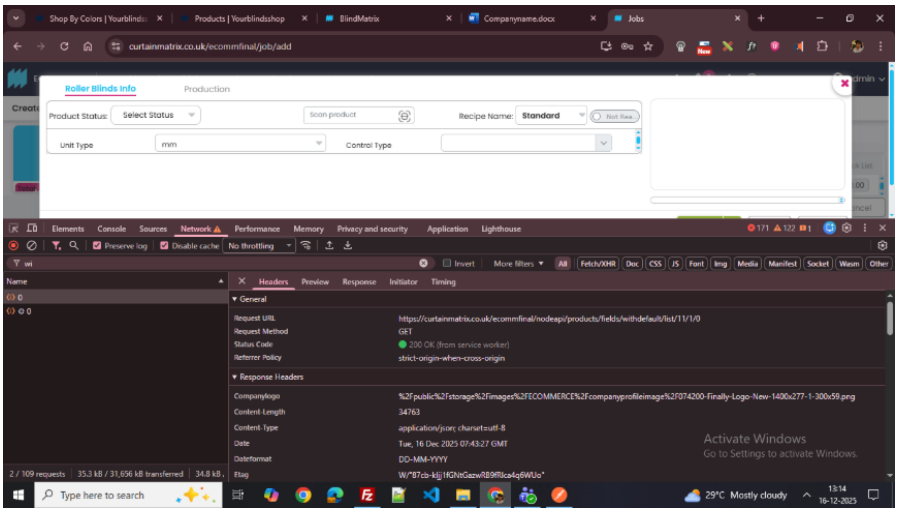
Returns ecommerce product details including recipe ID and configuration metadata.

URL Parameters

Name	Type	Description
productid	integer	Product ID

Response Fields

Field	Description
pei_productid	Product ID
Recipeid	Recipe ID used for configuration
pei_ecomProductName	Product name shown on ecommerce
pi_productdescription	Product description
pi_category	Product category (3 = fabric blinds, 4 = slat blinds)
pei_ecommercestatus	Ecommerce availability
pei_ecomFreeSample	Free sample enabled flag
pei_ecomsampleprice	Free sample price
minimum_price	Minimum product price



5. Product Configuration Schema

Get Field Schema

Endpoint

```
GET {nodeapipointer}/products/fields/withdefault/list/{recipeId}/1/0
```

Description

Retrieves the complete field schema used to configure the product.

URL Parameters

Name	Type	Description
recipeId	integer	Recipe ID from the 4.1 Get Product Details

Response

```
{
  "data": [ProductField],
  "netpricecomesfrom": 0,
  "costpricecomesfrom": 0
}
```

Important Field Type IDs

Purpose	Field Type ID
Unit	34
Supplier	17
Price Group	13
Width	7 / 8 / 11 / 31
Drop	9 / 10 / 12 / 32
Quantity	14
List	3
Fabric / Color	5 / 20 / 21

6. Option Filtering & Context Resolution

Option filtering determines **which options are valid and selectable** for a field based on the current product configuration context (price group, supplier, unit type, fabric, color, etc.).

This step is **mandatory before loading field options (Section 7)**.

6.1 Get Option Filters Based on General Data

Endpoint

```
POST {nodeapipointer}/products/fields/filterbasedongeneraldata
```

Description

This endpoint returns:

- The **allowed option IDs** for a given field
- The **allowed color IDs** (if applicable)
- Default **supplier** and **price group** selections (if defined by rules)

Filtering ensures that only **valid combinations** are shown to the user.

Headers

```
{
  "Accept": "application/json",
  "Content-Type": "application/json",
  "companyname": "{api_name}",
  "Ecommercekey": "{api_key}",
  "platform": "Ecommerce"
}
```

Request Body

```
{
  "fieldtypeid": 21,
  "productid": 101,
```

```
  "pricegroup": 3,
  "colorid": 0,
  "fabricid": 0,
  "unittype": 1,
  "customertype": 4,
  "optionid": [],
  "optioncolorids": [],
  "optionfabricids": [],
  "productionformulalist": [],
  "supplierid": null,
  "pricegrouparray": [],
  "fabriccolor": 0,
  "fieldid": 201,
  "level": 1
}
```

Request Body Parameters

Field	Type	Description
fieldtypeid	integer	Field type ID (e.g., 3, 5, 20, 21)
productid	integer	Product ID
pricegroup	integer	Selected price group ID
colorid	integer	Selected color ID (use `0` if none)
fabricid	integer	Selected fabric ID (use `0` if none)
unittype	integer	Unit type ID
customertype	integer	Ecommerce customer type (always 4)
optionid	array	Reserved for future use
optioncolorids	array	Reserved
optionfabricids	array	Reserved
productionformulalist	array	Reserved
supplierid	integer/null	Selected supplier ID
pricegrouparray	array	Reserved
fabriccolor	integer	Fabric/color context flag
fieldid	integer	Current field ID
level	integer	Field depth level

Response

```
{
  "data": {
    "optionarray": [1010, 1011, 1012],
    "coloridsarray": [],
    "selectsupplierid": 6,
    "selectpricegroupid": 3
  }
}
```

Response Fields

Field	Description
optionarray	Allowed option IDs for the field
coloridsarray	Allowed color IDs (if applicable)
selectsupplierid	Default supplier ID (if auto-selected)
selectpricegroupid	Default price group ID (if auto-selected)

6.2 Default Value Application Rules

If the response contains:

- `selectsupplierid`
- `selectpricegroupid`

Then:

- Apply them **automatically** if the user has not selected values
- Persist them for subsequent API calls (rules, pricing, options)

6.3 When to Call This API

This endpoint must be called:

- After loading the field schema
- When price group changes
- When supplier changes
- When fabric or color changes
- When unit type changes
- Before loading options (Section 7)

7. Load Field Options

Field options represent selectable values for **list, material, fabric, and color fields**.
Options are loaded dynamically based on the current configuration context.

This step must always be executed **after option filtering (Section 6)**.

7.1 Load Options for Select Fields

Endpoint

POST
{nodeapipointer}/products/get/fabric/options/list/{recipeId}/{level}/0/{fieldtype}/{fabriccolor}/{fieldid}

Description

Retrieves all selectable options for a given configuration field, filtered by the previously determined option IDs.

This endpoint supports:

- List fields
- Fabric fields
- Color fields
- Material fields

URL Parameters

Parameter	Type	Description

recipeId	integer	Recipe ID of the product
level	integer	Field depth level (top level = 1)
fieldtype	integer	Field type ID (e.g., 3, 5, 20, 21)
fabriccolor	integer	Fabric/Color context flag (use `0` if not required)
fieldid	integer	Field ID from schema

Headers

```
{
  "Accept": "application/json",
  "Content-Type": "application/json",
  "companyname": "{api_name}",
  "Ecommercekey": "{api_key}",
  "platform": "Ecommerce"
}
```

Request Body

```
{
  "customertype": 4,
  "filterids": [1010, 1011, 1012],
  "productionformulalist": [],
  "productid": 101
}
```

Request Body Parameters

Field	Type	Description
customertype	integer	Ecommerce customer type (always 4)
filterids	array	IDs returned from option filter API
productionformulalist	array	Reserved for production rules
productid	integer	Product ID

Response


```
{
  "data": [
    {
      "optionsvalue": [
        {
          "optionid": 1010,
          "optionname": "Fabric A",
          "optionimage": "",
          "fieldoptionlinkid": 9001,
          "availableForEcommerce": 1,
          "pricegroupid": 3,
          "optionid_pricegroupid": "1010_3"
        }
      ]}]}
```


Response Fields

Field	Description
optionid	Option ID
optionname	Display name
optionimage	Image URL (if available)
fieldoptionlinkid	Link ID used for subfield loading
availableForEcommerce	1 = selectable, 0 = ignore
pricegroupid	Associated price group
optionid_pricegroupid	Composite identifier

Mandatory Rules

- Ignore options where `availableForEcommerce = 0`
- Store `fieldoptionlinkid` for subfield loading
- Paginate results using page and perpage if more than 150 options exist

Pagination

```
?page=1&perpage=150
```

Repeat calls until all options are retrieved.

7.2 Option Selection Tracking (Local State)

Selected options must be tracked locally for **rules and pricing calculations**.

Required Structure

```
[
  {
    "optionvalue": 1010,
    "fieldtypeid": 21,
    "optionqty": 1,
    "fieldoptionlinkid": 9001,
    "fieldid": 201
  }
]
```

8 Subfield Loading (Dependent Fields)

Some options spawn **child fields** (subfields) that must be dynamically loaded and displayed.

8.1 Load Subfields

Trigger

This API must be called **when a selected option reports `subdatacount > 0`**.

Endpoint

`POST {nodeapipointer}/products/fields/list/0/{recipeId}/{level}/{fieldtype}/{masterparentfieldid}`

Description

Retrieves all dependent subfields associated with the selected parent option.

Subfields may themselves have options and further children.

URL Parameters

Parameter	Type	Description
recipeId	integer	Recipe ID
level	integer	Subfield depth level (start at 2)
fieldtype	integer	Parent field type ID
masterparentfieldid	integer	Parent field ID

Headers

```
{
  "Accept": "application/json",
  "Content-Type": "application/json",
  "companyname": "{api_name}",
```

```
  "Ecommercekey": "{api_key}",
  "platform": "Ecommerce"
}
```

Request Body

```
{
  "supplierid": 6,
  "productid": 101,
  "optionid": [1010],
  "subfieldoptionlinkid": [9001],
  "productionformulalist": [],
  "orderitemselectedvalues": {
    "201": [1010]
  }
}
```

Request Body Parameters

Field	Type	Description
supplierid	integer	Selected supplier ID
productid	integer	Product ID
optionid	array	Selected option ID(s)
subfieldoptionlinkid	array	Link IDs from option response
productionformulalist	array	Reserved
orderitemselectedvalues	object	Mapping of parent field → option

Response

```
{
  "data": [
    {
      "id": 301,
      "labelname": "Lining",
```

```
"fieldtypeid": 3,
"fieldlevel": 2,
"mandatory": 1 }]}
```

Response Fields

Field	Description
id	Subfield ID
labelname	Display label
fieldtypeid	Field type ID
fieldlevel	Depth level
mandatory	Required flag

Post-Processing Rules

- Insert returned fields into the working field list
- Immediately load options for subfields using **Section 7**
- Remove subfields when the parent option is deselected
- Maintain proper nesting inside `jsondata.subchild`

8.2 Nested Subfields

- Subfields may spawn additional child fields
- Increment level by +1 for each depth
- Repeat **7 → 8 → 7 → 8** until no further subfields exist

8.3 Subfield Removal Rules

When a parent option is changed:

- Remove all child subfields
- Clear their selected values
- Remove them from `jsondata.subchild`
- Recalculate rules and pricing

9. Dimension Validation & Fractions

Dimension validation ensures that user-entered width and drop values fall within the allowed manufacturing limits for the selected product, unit type, price group, fabric, and color.

9.1 Width & Drop Min / Max Validation

Endpoint

```
POST {nodeapipointer}/orderitems/check/widthdrop/minandmax/
```

Description

Validates the **minimum and maximum allowed values** for width and drop based on the selected configuration context.

This endpoint must be called:

- After selecting fabric/color
- After selecting unit type
- Before rules and pricing calculations
- Whenever width or drop values change

Headers

```
{
  "Accept": "application/json",
  "Content-Type": "application/json",
  "companyname": "{api_name}",
  "Ecommercekey": "{api_key}",
  "platform": "Ecommerce"
}
```

Request Body

```
{
  "width": "0",
  "drop": "0",
  "unittype": 1,
  "mode": "both",
  "pricegroup": 3,
  "colorid": "",
  "fieldtypeid": 21,
  "productid": 101
}
```

Request Body Parameters

Field	Type	Description
width	string	Current width value (use "0" for validation only)
drop	string	Current drop value (use "0" for validation only)
unittype	integer	Unit type ID (mm / inch)
mode	string	"both" validates width and drop
pricegroup	integer	Selected price group ID
colorid	string	Selected color ID (optional)
fieldtypeid	integer	Fabric or material field type ID
productid	integer	Product ID

Response

```
{
  "data": {
    "widthminmax": {
      "min": 300,
      "max": 3000
```

```
    },
    "dropminmax": {
      "min": 400,
      "max": 4000
    }
  }
}
```

Response Fields

Field	Description
widthminmax.min	Minimum allowed width
widthminmax.max	Maximum allowed width
dropminmax.min	Minimum allowed drop
dropminmax.max	Maximum allowed drop

9.2 Fraction Lists (Width / Drop Fractions)

Endpoint (General)

```
GET {apipointer}/public/api/appSetup/fractionlist/{product_id}
```

Endpoint (Unit-Specific)

```
GET {apipointer}/public/api/appSetup/fractionlist/{product_id}/-1/{unittype}
```

Description

Retrieves available fractional values for width and drop (primarily used for inch-based measurements).

Response

```
{
  "inchfractionselected": "0.125",
  "inchfraction": [
    { "value": "0", "text": "0/8" },
    { "value": "0.125", "text": "1/8" },
    { "value": "0.25", "text": "2/8" }
  ]
}
```

Usage Notes

- `inchfractionselected` represents the default fraction
- Fraction values must be populated into `jsondata.widthfraction` or `jsondata.dropfraction`
- The corresponding text must be set in `widthfractiontext` / `dropfractiontext`

10. VAT, Rules & Pricing

Pricing is calculated in **three sequential steps**:

1. Retrieve VAT percentage
2. Apply product rules
3. Calculate final price

Each step depends on the previous one.

10.1 Get VAT

Endpoint

- **URL**:{nodeapipointer}/job/get/vat/percentage/orderitem
- **Method**: POST

Description

Returns the VAT percentage applicable to the selected product.

Headers

```
{
  "Accept": "application/json",
  "Content-Type": "application/json",
  "companyname": "{api_name}",
  "Ecommercekey": "{api_key}",
  "platform": "Ecommerce"
}
```

Request Body

```
{
  "productid": 101
}
```

Response

```
{
  "data": {
    "vatpercentage": 20,
    "vatname": "VAT 20%"
  }
```

```
}
```

Response Fields

Field	Description
vatpercentage	VAT percentage
vatname	VAT display label

10.2 Apply Rules

Endpoint

- **URL**: *{nodeapipointer}/orderitems/calculate/rules*
- **Method**: POST

Description

Applies business rules and formulas to the configured product.

Rules may:

- Modify width or drop
- Enable or disable fields
- Adjust option pricing
- Calculate production material prices

Request Body (Key Fields)

```
{
  "vatpercentage": 20,
```

```
  "recipeid": 555,
  "productid": 101,
  "orderitemdata": [],
  "supplierid": 6,
  "mode": "pricetableprice",
  "width": "1200",
  "drop": "1800",
  "pricegroup": [3],
  "optiondata": [],
  "unittype": 1,
  "rulemode": 0,
  "widthfieldtypeid": 7,
  "dropfieldtypeid": 9,
  "fabricid": 1010,
  "colorid": 0
}
```

Request Body Parameters (Key)

Field	Description
orderitemdata	`jsondata` array in **rules mode**
optiondata	Selected options array
rulemode	`0` = update fields, `1` = production materials
mode	`"pricetableprice"` when applicable

Response

```
{
  "data": {
    "ruleresults": [
      {
        "fieldid": 7,
        "value": "1150"
      }
    ],
    "productionmaterialnetprice": 55.5
```

```
  }
}
```

Post-Processing Rules

- Apply ruleresults to your working field list
- Update width/drop values if modified
- Store production material prices if rulemode = 1

10.3 Calculate Price

Endpoint

- **URL**: *{nodeapipointer}/orderitems/calculate/option/price*
- **Method**: POST

Description

Calculates the **final net, VAT, and gross prices** based on applied rules and selected options.

Request Body (Key Fields)

```
{
  "productid": 101,
  "supplierid": 6,
  "mode": "pricetableprice",
  "width": "1150",
  "drop": "1800",
```

```
  "pricegroup": [3],
  "customertype": 4,
  "optiondata": [],
  "unittype": 1,
  "orderitemqty": 1,
  "vatpercentage": 20,
  "rulescostpricecomesfrom": 0,
  "rulesnetpricecomesfrom": 0,
  "fabricfieldtype": 21,
  "widthfieldtypeid": 7,
  "dropfieldtypeid": 9,
  "colorid": 0,
  "fabricid": 1010
}
```

Response

```
{
  "data": {
    "fullpriceobject": {
      "nettotal": 100,
      "gross": 120,
      "vatamount": 20
    },
    "currencysymbol": "£"
  }
}
```

Response Fields

Field	Description
nettotal	Net price
vatamount	VAT amount
gross	Gross price
currencysymbol	Currency symbol

11 jsondata Structure

11.1 Overview

jsondata is the **core configuration payload** used by the BlindMatrix Ecommerce platform.

It represents the **fully configured state of a product**, including:

- User-entered values (width, drop, quantity, text, numeric inputs)
- Selected options (fabric, color, list selections)
- Nested dependent fields (subfields)
- Fractional values for dimension fields
- Field hierarchy and mandatory rules

This structure is used by:

- Rules engine
- Pricing engine
- Free sample logic
- Order submission API

11.2 jsondata Format

jsondata is always an **array of objects**, where **each object represents one product field.**

Each object corresponds exactly to a field returned from the **Field Schema API.**

11.3 jsondata Field Object – Complete Definition


```
{
  "id": 7,
  "labelname": "Width",
  "value": "1200",
  "valueid": null,
  "type": 7,
  "optionid": null,
  "optionvalue": [],
  "optionquantity": null,
  "widthfraction": "0.125",
  "widthfractiontext": "1/8",
  "dropfraction": null,
  "dropfractiontext": null,
  "mandatory": 1,
  "fieldlevel": 1,
  "fieldname": "Width",
  "subchild": []
}
```


11.4 Field-Level Attribute Explanation

Attribute	Type	Description
id	integer	Field ID from schema
labelname	string	Label displayed to the user
value	string	Entered or selected value
valueid	integer / null	Internal value ID (used by specific field types)
type	integer	Field type ID
optionid	integer / null	Selected option ID (list/material fields)

optionvalue	array	Available options or selected option metadata
optionquantity	integer / null	Quantity for selected option (if applicable)
mandatory	integer	1 = required, 0 = optional
fieldlevel	integer	Field depth (1 = top-level)
fieldname	string	Backend field identifier
subchild	array	Nested child fields

11.5 Dimension-Specific Attributes (Width / Drop)

These attributes **apply only to width and drop fields.**

Attribute	Description
widthfraction	Decimal fraction value
widthfractiontext	Display fraction (e.g., 1/8)
dropfraction	Decimal fraction value
dropfractiontext	Display fraction

Example

```
{
  "value": "1200",
  "widthfraction": "0.125",
  "widthfractiontext": "1/8"
}
```

11.6 Option-Related Attributes

For **list, fabric, color, and material fields**:

Attribute	Description
optionid	Selected option ID
optionvalue	Option metadata (if required)
optionquantity	Quantity for the option

Example


```
{
  "id": 201,
  "labelname": "Fabric",
  "value": "1010",
  "optionid": 1010,
  "type": 21
}
```


11.7 Nested Fields (subchild)

Purpose

subchild stores **dependent fields** that are conditionally shown based on selected options.

Example


```
{
  "id": 201,
  "labelname": "Fabric",
  "subchild": [
    {
      "id": 301,
      "labelname": "Lining",
      "value": "Blackout",
      "type": 3,
```

```
      "fieldlevel": 2,
      "subchild": []
    }
  ]
}
```

Rules

- Only **active child fields** must be included
- Remove subfields when parent option changes
- Nest subfields recursively

11.8 Field Type Behavior in jsondata

Field Type	Behavior
Width / Drop	Uses value + fraction fields
Quantity	Uses value
Text	Uses value
Numeric	Uses value
List	Uses optionid
Fabric / Color	Uses optionid
Unit / Supplier / Price Group	Uses **valuename in rules mode**

11.9 Rules Mode vs Normal Mode

Normal Mode

- value and optionid use numeric IDs
- Used for UI and order submission

Rules Mode

Used for **rules calculation API.**

Special Handling Required:

Field Type	Expected Value
Unit (34)	valuename
Supplier (17)	valuename
Price Group (13)	valuename

11.10 Free Sample Mode

When preparing jsondata for **free sample pricing**:

Mandatory Adjustments

- Width = "0"
- Drop = "0"
- Retain fabric & color selection

11.11 Building jsondata – Step-by-Step

1. Start from field schema
2. Populate default values
3. Apply user selections
4. Attach fraction values
5. Insert subfields recursively
6. Apply rules updates
7. Final validation

11.12 Validation Rules

Before sending jsondata:

- All mandatory fields must be filled
- No orphan subfields allowed
- Width/Drop within min–max
- Correct mode applied (rules / free sample)

11.13 Example – Complete jsondata


```
[
  {
    "id": 7,
    "labelname": "Width",
    "value": "1150",
    "type": 7,
    "widthfraction": "0.125",
    "widthfractiontext": "1/8",
    "mandatory": 1,
    "fieldlevel": 1,
    "subchild": []
  },
  {
    "id": 201,
    "labelname": "Fabric",
    "value": "1010",
    "optionid": 1010,
    "type": 21,
    "mandatory": 1,
    "fieldlevel": 1,
    "subchild": [
      {
        "id": 301,
        "labelname": "Lining",
        "value": "Blackout",
        "type": 3,
        "fieldlevel": 2,
        "mandatory": 0,
        "subchild": []
      }
    ]}]
```

11.14 Where jsondata Is Used

API	Purpose
Rules	Modify fields & production cost
Pricing	Calculate final price
Free Sample	Zero-cost sample validation
Order Submit	Final order payload

11.15 One-Line Summary

jsondata is the **single source of truth** for product configuration in BlindMatrix Ecommerce, driving validation, pricing, rules, and order creation.

12. Order Submission

Order submission finalizes the transaction and creates the order in BlindMatrix.

12.1 Submit Order

Endpoint

- **URL**: *{apipointer}/public/api/ordersubmit*
- **Method**: POST

Description

Submits the completed order including billing details, pricing totals, and per-item configuration (jsondata).

Headers

```
{
  "Accept": "application/json",
  "Content-Type": "application/json",
  "companyname": "{api_name}",
  "Ecommercekey": "{api_key}",
  "platform": "Ecommerce"
}
```

Request Body (Structure)

```
{
  "billing_first_name": "ECOM",
  "billing_last_name": "Test",
  "billing_company": "Window Blinds Company",
  "billing_address_1": "No, 672 Temple Tower",
  "billing_city": "Chennai",
  "billing_country": "IN",
  "billing_state": "TN",
  "billing_postcode": "600035",
  "billing_email": "test@example.com",
  "billing_phone": "",
  "currency": "GBP",
  "payment_method": "stripe",
  "order_item_data": [
    {
      "ProductName": "Roller Blinds",
      "Quantity": 1,
      "jsondata": []
    }
```

```
  ],
  "order_total": "120.00",
  "deliverycost": 0
}
```

Mandatory Rules

- **Every key must be present**, even if empty
- Each item in order_item_data must contain its **own jsondata array**
- For multiple products, add multiple objects to order_item_data

Order Submission Flow Dependency

1. Pricing must be calculated
2. jsondata must be finalized
3. VAT must be included
4. Totals must match pricing API response

Retrieve the Job Details

Endpoint

***URL**: {apipointer}/public/api/getjobdetails*

- **Method**: GET

- **Headers**:
  - `Content-Type: application/json`
  - `companyname:` company name
  - `platform: Ecommerce`
  - `Ecommercekey:` get the secret key from the blindmatrix support team
  - `activity:{"ipaddress":"","location":"","devicenameversion":"","browsernameversion":""}`

Description:

Fetches the job details for a specified number of records with pagination support. The API returns a list of active job details, including customer information, job status, pricing, and more.

Query Parameters:

Parameter	Type	Required	Default	Description
`perpagecount`	integer	No	10	Number of job records to return per page
page	integer	No	1	Page number for pagination

Job Detail Fields in `result` Array

Field	Description
`id`	Job ID
`customerid`	Customer ID
`contactid`	Contact ID
`createddate`	Date the job was created
`jobref`	Job reference number
`jobstatus`	Job status (e.g., `Quote`, `Order`)
`accref`	Account reference

listprice	List price
netprice	Net price
vat	VAT amount
isvaton	VAT status (1 if VAT is applied)
deliverycost	Delivery cost
deliverycosttypeid	Delivery cost type ID
deliverycosttypename	Delivery cost type name
grossprice	Gross price
grossprofit	Gross profit
grossprofit_percentage	Gross profit percentage
costprice	Cost price
amountoutstanding	Amount outstanding
totpaid	Total amount paid
ready	Readiness status (0 = not ready, 1 = ready)
createdby	Username who created the job
createdat	Timestamp when job was created
updatedby	Username who last updated the job
job_payment_list	This field contains an array of all payment records associated with the job. Each entry includes details such as id, job_id, paid_date, payment_amount, payment_by, payment_type, and payment_method. If the job has received any payments, this field will contain one or more records; otherwise, it will be null or an empty array.
contact_address1	Customer address line 1
contact_address2	Customer address line 2
assignedto	Assigned user name
companyname	Company name
accounttype	Customer account type (e.g., Domestic, ECommerce)
contact_email	Customer email address
contact_firstname	Customer first name
contact_lastname	Customer last name
source	Job source (e.g., Facebook, Instagram)
contact_statecounty	Customer state or county
organizationname	Name of the organization linked to the job

Retrieve the order item details

Endpoint

- **URL**: *{apipointer}/public/api/getorderitems/{jobid}*
- **Method**: GET
- **Headers**:
  - Content-Type: application/json
  - companyname: company name
  - platform: Ecommerce
  - Ecommercekey: get the secret key from the blindmatrix support team
  - activity: {"ipaddress":"","location":"","devicenameversion":"","browsernamev ersion":""}

Description

Retrieves all order items for a specific job ID.

Request Parameters

Parameter	Type	Required	Description
jobid	integer	Yes	The ID of the job to retrieve items for

Response

Returns a 200 OK response with an array of order items.

Example Response


```
[
  {
    "id": 101,
    "jobid": 525,
    "productid": 204,
    "productname": "Roller Blind",
    "qty": 2,
    "seq": 1,
    "jsondata": "{...}",
    "description": "Living Room Window",
    "productiondate": "2025-05-15T00:00:00.000000Z",
    "orderitemcostprice": "10.000",
    "orderitemnetprice": "12.000",
    "vatprice": "2.400",
    "vatvalue": "20.000",
    "taxcategoryname": "Standard VAT",
    "vatonoff": 1,
    "grossprice": "14.400",
    "overrideprice": "0.000",
    "overridetype": 0,
    "overridevalue": null,
    "overridenetprice": "0.000",
    "overridevatprice": "0.000",
    "overridegrossprice": "0.000",
    "ready": 0,
    "onhold": 0,
    "suppliername": "Vision Blinds",
    "status": 0,
    "createdby": "admin",
    "createdat": "2025-05-14T12:30:00.000000Z"
  }
]
```


Field Descriptions

Field	Description
id	Order item ID

jobid	Job ID linked to this order
productid	ID of the product
productname	Name of the product
qty	Quantity ordered
seq	Custom sequence number for ordering
jsondata	Raw JSON of the order item (full config data)
description	Description of the item
productiondate	Production date
orderitemcostprice	Cost price of the item
orderitemnetprice	Net price (before VAT)
vatprice	VAT price
vatvalue	VAT percentage
taxcategoryname	Tax category (e.g. "Standard VAT")
vatonoff	1 = VAT on, 0 = off
grossprice	Gross price (Net + VAT)
overrideprice, overridetype, etc.	Override pricing fields if applicable
ready	0 = Not ready, 1 = Ready
onhold	0 = Not on hold, 1 = On hold
suppliername	Supplier name
createdby	Username of the creator
createdat	Timestamp of creation

Retrieve the job details and order item details

Endpoint

- **URL**: {apipointer}/public/api/fulljobgetdetails /{jobid}
- **Method**: GET
- **Headers**:
  - Content-Type: application/json
  - companyname: company name
  - platform: Ecommerce
  - Ecommercekey: get the secret key from the blindmatrix support team
  - activity: {"ipaddress":"","location":"","devicenameversion":"","browsernamev ersion":""}

Description

Retrieves all order items and job details for a specific job ID.

Request Parameters

Parameter		Type	Required	Description
jobid		integer	Yes	The ID of the job to retrieve items for

API Documentation for Appointment

Create Appointment API Flow Overview

Step 1: Retrieve appointment details using `getAppointmentDetails`.

Step 2: Use the selected appointmenttypeid, `date`, and `duration` to call `appointmentAvailableTime`.

Step 3: With the selected time slot from `appointmentAvailableTime`, proceed to create an appointment using `createAppointment.`

Step 1: Get Appointment Details

Endpoint

- **Live URL**: *https://blindmatrix.software/api/public/api/getappointmentdetails*
- **Method**: GET
- **Headers**:
  - Content-Type: application/json
  - companyname: company name (DB NAME)
  - platform: Ecommerce
  - Ecommercekey: get the secret key from the blindmatrix support team
  - activity:{"ipaddress":"","location":"","devicenameversion":"","browsernameversion":""}

Description:

This API endpoint retrieves appointment setup details including available products, calendar setup, sales sources, and appointment types.

Field Descriptions:

- Calendar settings (first day of week, work week, default duration)
- Available products list for appointment
- Appointment types with their appointment ids
- Sales source list

Step 2: Get Appointment Available time

Endpoint

- **Live URL**: *https://blindmatrix.software/api/public/api/appointmentavailabletime*
- **Method**: POST
- **Headers**:
  - Content-Type: application/json
  - companyname: company name (DB NAME)
  - platform: Ecommerce

- Ecommercekey: get the secret key from the blindmatrix support team
- activity: {"ipaddress":"","location":"","devicenameversion":"","browsernameversion":""}

Request Body

The request body should be a JSON object containing the following fields:

Field Name	Type	Description	Mandatory
appointmenttypeid	integer	Get the appointment type ID from the selected appointment in the Appointment Details API	Yes
date	Date	Date in YYYY-MM-DD format	Yes
duration	integer	Duration in minutes (optional). If not provided, calendar default duration is used.	No

Notes

- The system checks user working hours and existing appointments.
- Time slots are generated in increments of the provided or default duration.
- If `duration` is not provided in the request, it will be fetched from the default calendar setup.
- Time conflicts with existing appointments are avoided.
- Slots with more than one available user are grouped accordingly.

Step 3: Get Appointment Create Appointment

Endpoint

- **Live URL**: *https://blindmatrix.software/api/public/api/createappointment*
- **Method**: POST

- **Headers**:
  - `Content-Type: application/json`
  - `companyname:` company name (DB NAME)
  - `platform: Ecommerce`
  - `Ecommercekey: get the secret key from the blindmatrix support team`
  - `activity: {"ipaddress":"","location":"","devicenameversion":"","browsernamev ersion":""}`

Request Body

The request body must be a JSON object with the following fields:

Field Name	Type	Description	Mandatory
billing_company	String	Customer name (company)	Yes
billing_first_name	String	Customer first name	Yes
billing_last_name	String	Customer last name	No
billing_address_1	String	Address line 1	No
billing_address_2	String	Address line 2	No
billing_city	String	City	No
billing_state	String	State	No
billing_country	String	Country	No
billing_postcode	String	Postal Code	No
billing_email	String	Email Address	No
billing_phone	String	Phone Number	No
currency	String	Currency Code	No
api_url	String	URL for the API endpoint (default: https://blindmatrix.software)	Yes
job_data	JSON	Job-related custom data	No
source	String	Sale Source id for the appointment from Appointment details	Yes
contacttype	Integer	Contact type (default is 1) that is domestic	No

appointment_type	String	Appointment type ID selected from appointmentTypes	Yes
appointment_date	String	Date in YYYY-MM-DD format	Yes
appointment_time	String	Time in HH:MM format	Yes
appointment_users	String	User ID to assign (must be one of the users returned in availabletime api)	Yes
duration	Integer	Duration in minutes (optional). If not provided, calendar default duration is used.	No
appointment_status	String	The default status of the appointment is 'Inprogress', but it can be changed to 'Complete' or 'Ready'."	No
selectedProducts	String	List of products selected for the appointment from Appointment details	No
description	String	Notes or description for the appointment	No
timezone	String	Timezone while appointment created.	Yes

Format of `selectedProducts` JSON

```
[ {
   "id": "16",          // ID of the product
   "product_name": "Aluminium Venetians", // Name of the product
   "qty": "1"              // Quantity selected by the user
 },{
   "id": "18",
   "product_name": "Blackout Roller Blinds",
   "qty": "3"
 }]
```

Timezone Detection Suggestion

To dynamically set the timezone from the user's browser:

```
timezone = Intl.DateTimeFormat().resolvedOptions().timeZone;
```

This will return values such as `"Asia/Kolkata"`, which are compatible with the API.

Notes

- `userid` used here should match the one selected in `appointmentTypes` from `getappointmentdetails`.
- For createAppointment, you must pass the duration for the selected time slot, as it determines how much time will be blocked on the calendar. If the duration is not provided in the 'Get Appointment Available Time' API or not sent at all, the calendar's default duration will be used.
- All product selections and job data are embedded in the payload.