

# Developer guidelines

Easy PV and Heatpunk have a number of API endpoints that can be used to integrate CRMs, job management platforms and other applications. Find out how here!

- [Getting started](#)
- [Your first call - create a project](#)
- [Interacting with forms and files](#)
- [Get project details, bill of materials and order links](#)

# Getting started

Our API for Easy PV and Heatpunk is open, and we welcome developers working on new integrations for CRMs, job management platforms and other applications. If you have an application that you would like to connect please give us a shout - we'd be really happy to help.

## Sandbox access

If you are developing an integration with another application we can give you access to our sandbox environment. Please ask for details. In all the examples we give here we use the live API at the <https://easy-pv.co.uk/> domain; to use the Easy PV sandbox simply change this to <https://sandbox.easy-pv.co.uk>.

We use the same underlying platform for Heatpunk, and almost all the endpoints we refer to here will also work for Heatpunk (we'll make clear if not). For Heatpunk, use <https://heatpunk.co.uk> for the live environment, and <https://sandbox.heatpunk.co.uk> for the sandbox.

## Documentation and testing

Documentation for the API is at <https://easy-pv.co.uk/api/about/> and <https://heatpunk.co.uk/api/about/>.

You can test calls from the documentation itself. It will also give example CURL calls which you can test from a command line. Platforms such as [Postman](#) or [Hoppscotch](#) are also very useful for development of an API integration.

## Generating an API key

Most API endpoints that we make publicly available can only be accessed through a 'pro' team API key. For 'Pro' team owners and admins please login and go to your 'Pro account settings' dashboard, then click on the 'CRM connections' tab. Click on the 'Generate key' button to create an API key.

It is very important that you keep the API key secure. Anyone who gains access to it will be able to access and change all data for the users in your team, including customer names and addresses. We recommend it is only used for server-server requests, and stored as an environment variable and kept out of git repositories. If you wish to keep a backup of the key, use a secure password manager.

Note that we don't keep a record of the key. If you lose it, you will need to generate a new one for your application.

If you are developing an application just for one company, you will only need to generate one key. If you have a platform that many companies use, every company will need to generate their own key and save it to their account on your platform.

# Your first call - create a project

/projects/create

You can create a new project using the [projects/create](#) endpoint.

## Constructing your API request

All projects must have an `owner`, which must be specified in the request POST. The owner email must be a member of the pro team that owns the API key. The owner can then subsequently share the project with other members of the team if they wish.

Additional information needs to be included in the `meta` object. This can include: `projectName`, `customerName`, `customerEmail`, `customerPhone`, `address`, `postcode`, `crmReference`, `projectType`

`crmReference` can be used for your internal reference - it's likely that you will have a unique database ID that you want to be associated with the project. All fields are optional.

`projectType` can be used to set the design mode for the project ( `three-d-model`, `quick-roof`, `roof-outline` ). For 3D projects you can also enable automatic roof scanning functionality by including `"magicMode" : "true"`.

You can run this from the command line or using a tool like postman to try it out. Change the owner email to a user within the Easy PV team (probably your own if you want to be able to check it has been successful), and put your key in the X-API-KEY header.

```
curl -X 'POST'  
'https://sandbox.easy-pv.co.uk/api/v1/projects/create'  
-H 'accept: application/json'  
-H 'X-API-KEY: *****KEY*****'  
-H 'Content-Type: application/json' -d '{  
  "owner": "sales@midsummerenergy.co.uk",  
  "meta": {  
    "customerName": "Joe Bloggs"  
  }  
}'
```

The endpoint will return a JSON object with a `projectId` property. You will almost certainly want to save this `projectId` within your own database.

You can link directly to a project with a link of the following format:

<https://easy-pv.co.uk/?project=872019>

Congratulations! You've used our API to create a project within Easy PV. Read on to see how you can retrieve files and form data from an Easy PV project.

# Interacting with forms and files

Within Easy PV and Heatpunk, users complete forms and generate PDF reports. In your application you may want to view the completed files and forms associated with a project, and fetch completed form data and PDF documents.

## To list forms for a project:

To get a list of completed forms for a project, use the `/api/v1/projects/forms/list` endpoint. Provide `projectId` and `userEmail` as GET parameters. `userEmail` is necessary as the API key is shared amongst all members of your team, but projects aren't necessarily shared with all members of the team. When we receive the request we will check if the given email has the right to access the project.

```
curl -X 'GET' \
  'https://sandbox.easy-
pv.co.uk/api/v1/projects/forms/list?projectId=872034&userEmail=sales%40midsummerenergy.co.uk' \
  -H 'accept: application/json' \
  -H 'X-API-KEY: ---KEY---'
```

This should return an object with completed forms and surveys.

```
{
  "forms": [
    {
      "id": "customerProposal"
    },
    {
      "id": "enaConnect"
    },
    {
      "id": "f08_v2"
    },
    {
      "id": "g99_50kw"
    },
    {
      "id": "scaffoldingRequest"
    },
    {
      "id": "g98"
    }
  ]
}
```

```

    },
    {
      "id": "g99_A1_1"
    },
    {
      "id": "g99_A1_2"
    },
    {
      "id": "letterOfConsent"
    },
    {
      "id": "install_record"
    },
    {
      "id": "g99_A3_1"
    },
    {
      "id": "g99_A3_2"
    }
  ],
  "surveys": [
    {
      "id": "survey"
    }
  ]
}

```

## To retrieve form fields:

Use `/api/v1/projects/forms/form.`

You will need to provide the ID of the project, ID of the form (from the response above), and the userEmail. Here's an example:

```

curl -X 'GET' \
  'https://sandbox.easy-
pv.co.uk/api/v1/projects/forms/form?projectId=872034&form=letterOfConsent&userEmail=andy%40midsummer
energy.co.uk&formCategory=forms' \
  -H 'accept: application/json' \

```

```
-H 'X-API-KEY: ---KEY---
```

## To list files saved to a project:

Use `/api/v1/projects/files/list`. Set the type property to projects, and provide the ID of the project and the userEmail as before. Files saved within Easy-PV are listed within the `fileRegistry.uploads` array of the response.

```
curl -X 'GET' \  
  'https://sandbox.easy-  
pv.co.uk/api/v1/files/list?type=projects&id=872034&userEmail=andy%40midsummerenergy.co.uk' \  
  -H 'accept: application/json' \  
  -H 'X-API-KEY: ---KEY---
```

## To download a file:

Use `/api/v1/files/file`.

Provide the same details as the previous request, plus the file category (uploads) and the file name.

```
curl -X 'GET' \  
  'https://sandbox.easy-  
pv.co.uk/api/v1/files/file?type=projects&id=872034&userEmail=andy%2Bjumptech%40midsummerenergy.co.uk  
&fileCategory=uploads&file=test%20project%20%20letter%20of%20consent~Letter%20of%20Consent.pdf&res  
ponseFormat=b64' \  
  -H 'accept: application/json' \  
  -H 'X-API-KEY: ---KEY---
```

Setting responseFormat to either `b64` or `meta` will include a meta object in the response. This includes a datetime string for when the file was created and last updated. `b64` will also return the document as a base64-encoded string in the response. `raw` will just return the file as stored.

We have updated the attribute used by the API when specifying the email address of the user that owns the record so it is consistent across all our endpoints. We now always refer to this attribute as `userEmail`. If you previously used `ownerEmail` this will still work as it has been set up as an alias of `userEmail`.



# Get project details, bill of materials and order links

Easy PV and Heatpunk generate a full bill of materials for a project, and for our UK and Ireland sites you can very quickly place an order for the components from Midsummer. So you may like to import the shopping cart into your application and display an ordering link. Users love the ease of ordering all the kit for an installation.

The cart is based on the bill of materials generated by the software for a given project. The bill of materials for the project is refreshed every time the overview page of the project is opened by a user in Easy PV or Heatpunk.

To fetch project data, including the cart and order link, use the `api/v1/projects/data` endpoint. You need to pass in the project ID, and the user you are acting as. The user must have view rights to the project.

## Sample request:

```
curl --location 'https://easy-pv.co.uk/api/v1/projects/data?projectId=[YOUR PROJECT ID]&userEmail=[YOUR USER EMAIL]' \
--header 'accept: application/json' \
--header 'X-API-KEY: [YOUR API KEY]'
```

## What to expect in the response:

You will get back an object which includes `projectData`, a `cart` array and a `midsummerOrderLink` string.

### ProjectData

The `projectData` array will include core project and customer information. For example:

```
"projectData": {
  "lat": 52.2305742,
  "lng": 0.14036,
  "zoom": 19,
  "address": "101 Kings Hedges Rd, Cambridge , UK",
  "postcode": "CB4 2QD",
  "dateCreated": "2024-10-02 17:42:45",
  "dateModified": "2024-10-17 14:11:58",
  "projectName": "Sample Project",
```

```
"customerName": "Testing Testington",
"includeTerrain": false,
"MCSSiteVisitDisclaimer": false,
"geography": {
  "altitude": 12,
  "distance": "2to20",
  "hillSlope": null,
  "hillZone": null,
  "terrain": "country",
  "topography": "no",
  "windZone": 3,
  "snowZone": 2
},
"projectType": "three-d-model",
"magicMode": true,
"customerPhone": "0777 777 777",
"customerEmail": "testing@testington.com",
"userID": "19380",
"status": "Lead",
"mapImagery": {
  "selectedType": "orthorectified",
  "orthorectified": {
    "fileId": "rG2G25",
    "rotationUsed": 2.2618198,
    "radius": 48.137703120171004
  }
},
"projectID": "986206",
"MCSDisclaimer": true,
"MCSReservedCapacity": 50,
"MCSAdditionalConsumption": 0,
"MCSConsumption": "3500",
"MCSOccupancy": "home all day",
"totalConsumption": 3500,
"selfConsumption": 980.7616984962381,
"threeDModelSettingsViewDirection": [
  -4.3836904133806645,
  -6.107308078177546,
  10.644089349920092,
  0.054191298862051804,
```

```
2.2301402617071955,  
2.241462631675495  
],  
"threeDSnapshots": [  
  "l6jskm"  
],  
}
```

## Cart

The `cart` array contains the full bill of materials as a JSON object. This contains details of all the products that have been specified in the project. For example:

```
"cart": [  
  {  
    "name": "Jinko Tiger Neo 440W N-Type All Black Mono solar panel",  
    "qty": 6,  
    "item": 84,  
    "line": 504,  
    "dbID": "5265",  
    "availableFrom": {  
      "midsummer": {  
        "ID": "5265",  
        "price": 504  
      }  
    }  
  },  
  {  
    "name": "Growatt MIN 10000 TL-X2 Three MPPT 1ph inverter",  
    "qty": 1,  
    "item": 1050,  
    "line": 1050,  
    "dbID": "5397",  
    "availableFrom": {  
      "midsummer": {  
        "ID": "5397",  
        "price": 1050  
      }  
    }  
  },  
  {
```

```

    "name": "Giv.AC 3.0 inverter",
    "qty": 1,
    "item": 933.8,
    "line": 933.8,
    "dbID": "3040",
    "availableFrom": {
      "midsummer": {
        "ID": "3040",
        "price": 933.8
      }
    }
  },
  {
    "name": "GivEnergy 5.2kWh LiFePO4 Battery",
    "qty": 1,
    "item": 1796.2,
    "line": 1796.2,
    "dbID": "4284",
    "availableFrom": {
      "midsummer": {
        "ID": "4284",
        "price": 1796.2
      }
    }
  }
],
[

```

**midsummerOrderLink**

The `midsummerOrderLink` is a URL that can be used to order everything needed for a project from Midsummer. For example:

"midsummerOrderLink":  
"https://midsummerwholesale.co.uk/upload/?order=eyJpdGVtcyl6eyl3OCt6eyJ0aXRzZSI6IkxhYmVsIHNoZWV0IiwicXRS5joyfSwiMzgwlj7InRpdGxlIjoMTAwbSByZWVslG9mIDZtbTIgc29sYXIgY2FibGUlLCJxdHkiOjF9LClxNzc5lj7InRp dGxlIjoiTUM0IDZtbSBDb25uZWN0b3IgUGFpciIsbnF0eSI6NH0sljlYTgiOnsidGI0bGUIOiJuaWdvIFJldHJvZml0IEZyYW l1IE1vdW50ZWQgT3B0aWwpc2VyIFRTNC1BLU8iLCJxdHkiOjZ9LClYNTg2lj7InRpdGxlIjoIRmFzdGVuc29sIHJhaWwg c3BsawNWlliwicXRS5Jo2fSwiMjU4NyI6eyJ0aXRzZSI6IkZhcn3RIbnNvbCBByYWlsIGJybHQiLCJxdHkiOjZ9LClYNTk0lj7InRp dGxlIjoIRmFzdGVuc29sIHBCnRyYWI0IGZsYXQgdGlSZSByb29mIGHvb2siLCJxdHkiOjI0fSwiMjgxMil6eyJ0aXRzZSI6Ik JhdHRlcnckgSGF6YXJklFdchcm5pbmcgTGFiZWwgUGFjayIsbnF0eSI6MX0sljl4MTQiOnsidGI0bGUIOiIlqKk5FVCoiEVtbGI

0ZSBCaS1kaXJlY3Rpb25hbCBnZXRLciBFQ0EyLm4qIiwicXR5IjoxfSwiMzA0MCI6eyJ0aXRzZSI6Ikdpd15BQyAzLjAgaW52ZXJ0ZXliLCJxdHkiOjF9LCIzMDQ2Ijpb7InRpdGxlljoiR2I2RW5lcmd5IFdpRmkkgRG9uZ2xlliwicXR5IjoxfSwiMzE1NyI6eyJ0aXRzZSI6Ikhzc3RlbnNvbCBibGFjayB1bml2ZXJzYWwgY2xhbXAiLCJxdHkiOjE4fSwiMzU4NCI6eyJ0aXRzZSI6Ikd5b3dhhdHQgU2hpbmVXaUZpLVgiLCJxdHkiOjF9LCIzNjU1Ijpb7InRpdGxlljoiR2I2RW5lcmd5IERDIE1pbmlhdHVyZSBDaXJjdWl0IEJyZWFrZXIgcKE1DQikiLCJxdHkiOjJ9LCI0Mjg0Ijpb7InRpdGxlljoiR2I2RW5lcmd5IDUuMmtXaCBMaUZiUE80IEJhdHRlcniLCJxdHkiOjF9LCI0NDg0Ijpb7InRpdGxlljoiR2I2RW5lcmd5IFBsdWcgdG8gTHVnIENhYmxlChHZW4zKSIsInF0eSI6MX0sljQ0OTMiOnd5IGl0bGUiOjBQYBpc29sYXRvciAtIEINTyAtIDlwQSA0LXBvbGUiLCJxdHkiOjJ9LCI0NTU1Ijpb7InRpdGxlljoiQUMgaXNvbGF0b3IgLSBjTU8gLSA2M0EgNC1wb2xlliwicXR5IjoxfSwiNTEwNSI6eyJ0aXRzZSI6IjY1YmJlciBHcm9tbWV0IGZvciBHZW4zIEludmVydGVyIFBsdWcgQ2FibGVzIiwicXR5IjoxfSwiNTI2NSI6eyJ0aXRzZSI6IkpmbmtvIFRpZ2VyeIE5lbyA0NDBXIE4tVHlwZSBbbGwgQmxhY2sgTW9ubyBzb2xhciBwYW5lbnF0eSI6Nn0sljUyNzYiOnd5IGl0bGUiOjBQYBpc29sYXRvciAtIEINTyAtIDlwQSA0LXBvbGUiLCJxdHkiOjJ9LCI0NTU1Ijpb7InRpdGxlljoiQUMgaXNvbGF0b3IgLSBjTU8gLSA2M0EgNC1wb2xlliwicXR5IjoxfSwiNTEwNSI6eyJ0aXRzZSI6IjY1YmJlciBHcm9tbWV0IGZvciBHZW4zIEludmVydGVyIFBsdWcgQ2FibGVzIiwicXR5IjoxfSwiNTU2MSI6eyJ0aXRzZSI6IkdpdVXJneSBHRU0xMjBDVCBnb2RidXMgRW5lcmd5IE1ldGVyIiwicXR5IjoxfSwiIjpb7InRpdGxlljoiTWF0dCBTY2FmZm9sZGluZyBvdmdVylGNvbnNlcnZhdG9yeSIsInF0eSI6MX0sljQ0NDg0Ijpb7InRpdGxlljoiQUMgaXNvbGF0b3IgLSBjTU8gLSA2M0EgNC1wb2xlliwicXR5IjoxfSwiNTEwNSI6eyJ0aXRzZSI6Ikhzc3RlbnNvbCBibGFjayB1bml2ZXJzYWwgY2xhbXAiLCJxdHkiOjE4fSwiMzU4NCI6eyJ0aXRzZSI6Ikd5b3dhhdHQgU2hpbmVXaUZpLVgiLCJxdHkiOjF9LCIzNjU1Ijpb7InRpdGxlljoiR2I2RW5lcmd5IERDIE1pbmlhdHVyZSBDaXJjdWl0IEJyZWFrZXIgcKE1DQikiLCJxdHkiOjJ9LCI0Mjg0Ijpb7InRpdGxlljoiR2I2RW5lcmd5IDUuMmtXaCBMaUZiUE80IEJhdHRlcniLCJxdHkiOjF9LCI0NDg0Ijpb7InRpdGxlljoiR2I2RW5lcmd5IFBsdWcgdG8gTHVnIENhYmxlChHZW4zKSIsInF0eSI6MX0sljQ0OTMiOnd5IGl0bGUiOjBQYBpc29sYXRvciAtIEINTyAtIDlwQSA0LXBvbGUiLCJxdHkiOjJ9LCI0NTU1Ijpb7InRpdGxlljoiQUMgaXNvbGF0b3IgLSBjTU8gLSA2M0EgNC1wb2xlliwicXR5IjoxfSwiNTEwNSI6eyJ0aXRzZSI6IjY1YmJlciBHcm9tbWV0IGZvciBHZW4zIEludmVydGVyIFBsdWcgQ2FibGVzIiwicXR5IjoxfSwiNTI2NSI6eyJ0aXRzZSI6IkpmbmtvIFRpZ2VyeIE5lbyA0NDBXIE4tVHlwZSBbbGwgQmxhY2sgTW9ubyBzb2xhciBwYW5lbnF0eSI6Nn0sljUyNzYiOnd5IGl0bGUiOjBQYBpc29sYXRvciAtIEINTyAtIDlwQSA0LXBvbGUiLCJxdHkiOjJ9LCI0NTU1Ijpb7InRpdGxlljoiQUMgaXNvbGF0b3IgLSBjTU8gLSA2M0EgNC1wb2xlliwicXR5IjoxfSwiNTEwNSI6eyJ0aXRzZSI6IjY1YmJlciBHcm9tbWV0IGZvciBHZW4zIEludmVydGVyIFBsdWcgQ2FibGVzIiwicXR5IjoxfSwiNTI2NSI6eyJ0aXRzZSI6IkpmbmtvIFRpZ2VyeIE5lbyA0NDBXIE4tVHlwZSBbbGwgQmxhY2sgTW9ubyBzb2xhciBwYW5lbnF0eSI6Nn0sljUyNzYiOnd5IGl0bGUiOjBQYBpc29sYXRvciAtIEINTyAtIDlwQSA0LXBvbGUiLCJxdHkiOjJ9LCI0NTU1Ijpb7InRpdGxlljoiQUMgaXNvbGF0b3IgLSBjTU8gLSA2M0EgNC1wb2xlliwicXR5IjoxfSwiNTEwNSI6eyJ0aXRzZSI6IjY1YmJlciBHcm9tbWV0IGZvciBHZW4zIEludmVydGVyIFBsdWcgQ2FibGVzIiwicXR5IjoxfSwiNTI2NSI6eyJ0aXRzZSI6IkpmbmtvIFRpZ2VyeIE5lbyA0NDBXIE4tVHlwZSBbbGwgQmxhY2sgTW9ubyBzb2xhciBwYW5lbnF0eSI6Nn0sljUyNzYiOnd5IGl0bGUiOjBQYBpc29sYXRvciAtIEINTyAtIDlwQSA0LXBvbGUiLCJxdHkiOjJ9LCI0NTU1Ijpb7InRpdGxlljoiQUMgaXNvbGF0b3IgLSBjTU8gLSA2M0EgNC1wb2xlliwicXR5IjoxfSwiNTEwNSI6eyJ0aXRzZSI6IjY1YmJlciBHcm9tbWV0IGZvciBHZW4zIEludmVydGVyIFBsdWcgQ2FibGVzIiwicXR5IjoxfSwiNTI2NSI6eyJ0aXRzZSI6IkpmbmtvIFRpZ2VyeIE5lbyA0NDBXIE4tVHlwZSBbbGwgQmxhY2sgTW9ubyBzb2xhciBwYW5lbnF0eSI6Nn0sljUyNzYiOnd5IGl0bGUiOjBQYBpc29sYXRvciAtIEINTyAtIDlwQSA0LXBvbGUiLCJxdHkiOjJ9LCI0NTU1Ijpb7InRpdGxlljoiQUMgaXNvbGF0b3IgLSBjTU8gLSA2M0EgNC1wb2xlliwicXR5IjoxfSwiNTEwNSI6eyJ0aXRzZSI6IjY1YmJlciBHcm9tbWV0IGZvciBHZW4zIEludmVydGVyIFBsdWcgQ2FibGVzIiwicXR5IjoxfSwiNTI2NSI6eyJ0aXRzZSI6IkpmbmtvIFRpZ2VyeIE5lbyA0NDBXIE4tVHlwZSBbbGwgQmxhY2sgTW9ubyBzb2xhciBwYW5lbnF0eSI6Nn0sljUyNzYiOnd5IGl0bGUiOjBQYBpc29sYXRvciAtIEINTyAtIDlwQSA0LXBvbGUiLCJxdHkiOjJ9LCI0NTU1Ijpb7InRpdGxlljoiQUMgaXNvbGF0b3IgLSBjTU8gLSA2M0EgNC1wb2xlliwicXR5IjoxfSwiNTEwNSI6eyJ0aXRzZSI6IjY1YmJlciBHcm9tbWV0IGZvciBHZW4zIEludmVydGVyIFBsdWcgQ2FibGVzIiwicXR5IjoxfSwiNTI2NSI6eyJ0aXRzZSI6IkpmbmtvIFRpZ2VyeIE5lbyA0NDBXIE4tVHlwZSBbbGwgQmxhY2sgTW9ubyBzb2xhciBwYW5lbnF0eSI6Nn0sljUyNzYiOnd5IGl0bGUiOjBQYBpc29sYXRvciAtIEINTyAtIDlwQSA0LXBvbGUiLCJxdHkiOjJ9LCI0NTU1Ijpb7InRpdGxlljoiQUMgaXNvbGF0b3IgLSBjTU8gLSA2M0EgNC1wb2xlliwicXR5IjoxfSwiNTEwNSI6eyJ0aXRzZSI6IjY1YmJlciBHcm9tbWV0IGZvciBHZW4zIEludmVydGVyIFBsdWcgQ2FibGVzIiwicXR5IjoxfSwiNTI2NSI6eyJ0aXRzZSI6IkpmbmtvIFRpZ2VyeIE5lbyA0NDBXIE4tVHlwZSBbbGwgQmxhY2sgTW9ubyBzb2xhciBwYW5lbnF0eSI6Nn0sljUyNzYiOnd5IGl0bGUiOjBQYBpc29sYXRvciAtIEINTyAtIDlwQSA0LXBvbGUiLCJxdHkiOjJ9LCI0NTU1Ijpb7InRpdGxlljoiQUMgaXNvbGF0b3IgLSBjTU8gLSA2M0EgNC1wb2xlliwicXR5IjoxfSwiNTEwNSI6eyJ0aXRzZSI6IjY1YmJlciBHcm9tbWV0IGZvciBHZW4zIEludmVydGVyIFBsdWcgQ2FibGVzIiwicXR5IjoxfSwiNTI2NSI6eyJ0aXRzZSI6IkpmbmtvIFRpZ2VyeIE5lbyA0NDBXIE4tVHlwZSBbbGwgQmxhY2sgTW9ubyBzb2xhciBwYW5lbnF0eSI6Nn0sljUyNzYiOnd5IGl0bGUiOjBQYBpc29sYXRvciAtIEINTyAtIDlwQSA0LXBvbGUiLCJxdHkiOjJ9LCI0NTU1Ijpb7InRpdGxlljoiQUMgaXNvbGF0b3IgLSBjTU8gLSA2M0EgNC1wb2xlliwicXR5IjoxfSwiNTEwNSI6eyJ0aXRzZSI6IjY1YmJlciBHcm9tbWV0IGZvciBHZW4zIEludmVydGVyIFBsdWcgQ2FibGVzIiwicXR5IjoxfSwiNTI2NSI6eyJ0aXRzZSI6IkpmbmtvIFRpZ2VyeIE5lbyA0NDBXIE4tVHlwZSBbbGwgQmxhY2sgTW9ubyBzb2xhciBwYW5lbnF0eSI6Nn0sljUyNzYiOnd5IGl0bGUiOjBQYBpc29sYXRvciAtIEINTyAtIDlwQSA0LXBvbGUiLCJxdHkiOjJ9LCI0NTU1Ijpb7InRpdGxlljoiQUMgaXNvbGF0b3IgLSBjTU8gLSA2M0EgNC1wb2xlliwicXR5IjoxfSwiNTEwNSI6eyJ0aXRzZSI6IjY1YmJlciBHcm9tbWV0IGZvciBHZW4zIEludmVydGVyIFBsdWcgQ2FibGVzIiwicXR5IjoxfSwiNTI2NSI6eyJ0aXRzZSI6IkpmbmtvIFRpZ2VyeIE5lbyA0NDBXIE4tVHlwZSBbbGwgQmxhY2sgTW9ubyBzb2xhciBwYW5lbnF0eSI6Nn0sljUyNzYiOnd5IGl0bGUiOjBQYBpc29sYXRvciAtIEINTyAtIDlwQSA0LXBvbGUiLCJxdHkiOjJ9LCI0NTU1Ijpb7InRpdGxlljoiQUMgaXNvbGF0b3IgLSBjTU8gLSA2M0EgNC1wb2xlliwicXR5IjoxfSwiNTEwNSI6eyJ0aXRzZSI6IjY1YmJlciBHcm9tbWV0IGZvciBHZW4zIEludmVydGVyIFBsdWcgQ2FibGVzIiwicXR5IjoxfSwiNTI2NSI6eyJ

## Customise cart and order links

If you wish to modify the bill of materials for a project and create your own `cart` object, you can use this to generate a custom order link. If you do this, you can also add some additional parameters such as a delivery address or order reference to the order (see instructions in the next section).

In its simplest form, an order can be just an object with an 'items' property. For each item, specify the the products by providing the product ID and the quantity required.

```
const order = {
  "items": {
    "2586": {
      "qty": 99
    },
    "4258": {
      "qty": 99
    }
  }
}
```

Here is a more complex order, with a shipping address, despatch date and reference:

```
const order = {
  "items": {
    "2586": {
```

```
        "qty": 99
      },
    },
    "deliveryOption": "standard",
    "shippingAddress": {
      "postcode": "CB24 6AZ",
      "add1": "Midsummer Energy",
      "add2": "Cambridge Road Industrial Estate",
      "add3": "Milton",
      "add4": "",
      "email": "sales@midsummerenergy.co.uk",
      "phone": "01223 858414",
      "firstname": "Andy",
      "lastname": "Rankin"
    },
    "despatchDate": "2023-02-08",
    "reference": "trial order"
  }
}
```

Once you have created the JSON object containing the products required for the project you can create an order string by following these steps:

1. Stringify the object
2. Base64 encode the result of the stringified object
3. URL encode the base64 encoded string
4. Append your string to the following URL to create your custom order link

```
https://midsummerwholesale.co.uk/upload?order=[result string]
```

In JavaScript the function you need to generate an order is:

```
function createOrderLink(order) {
  return
  "https://midsummerwholesale.co.uk/upload?order="+encodeURIComponent(btoa(JSON.stringify(order)))
}
```