BLOUK EDI Order File Generator

A browser-based tool for Bloomsbury Publishing that converts subscription order data from Excel into the fixed-width EDI format required by the Taylor & Francis / BLOUK distribution system, and separately generates XML metadata files from a streamlined three-column spreadsheet.

No installation, server, or build tools required — the entire application runs in the browser from a single folder.

Table of Contents

Overview

The application provides two independent tools in a single interface:

EDI Order File Generator — accepts a 16-column order spreadsheet and produces a plain-text fixed-width EDI file in the BLOUK T1 format, ready to submit to the Taylor & Francis / BLOUK distribution system.

XML Metadata Generator — accepts a simplified 3-column spreadsheet (ISSN, Title, Page Extent) and produces one XML metadata file per row. All remaining specification values — trim size, binding, lamination, paper type, and spine width — are derived automatically from the page extent using defined business rules. The XML files are bundled into `metadata.zip`, and a plain-text summary report (`metadata_summary.txt`) is downloaded alongside it.

Both tools enforce strict column validation on upload: if the uploaded file does not exactly match the expected template structure, the file is rejected before any data is processed and a detailed per-column error report is shown.

File Structure

```
blouk_edi/
├── index.html              — Application markup
├── style.css               — All visual styling and layout
├── script.js               — All application logic
├── order_file.xlsx         — Order upload template (download from app)
├── metadata_template.xlsx  — Metadata upload template (download from
app)
└── README.md               — This document
```

All logic is in `script.js`; all presentation is in `style.css`; `index.html` contains only semantic structure. The two `.xlsx` template files must remain in the same root directory as `index.html` for the in-app template download buttons to work.

Quick Start

1. Copy the entire `blouk_edi/` folder to any location on your machine or a web server.
2. Open `index.html` in a modern web browser (Chrome, Edge or Firefox recommended).
3. An internet connection is required on first load to fetch CDN libraries. After that, the browser cache will serve them offline.

EDI Order File Generator

How to Use

Step 1 — Download the template (first time only) Click **Download Order Template** in the Import Excel Data panel to save `order_file.xlsx`. Use this file as the basis for every upload — do not rename, reorder or add columns.

Step 2 — Populate the template Fill in your subscription order rows below the header row. Each row represents one journal issue or subscription line. See Order Template — Column Reference for field details.

Step 3 — Upload the file Drag `order_file.xlsx` onto the upload zone or click to browse. The app validates that the column structure exactly matches the template. If validation passes, a green confirmation shows the row count and the Source Data tab populates with a preview table.

Step 4 — Adjust File Settings (if required) Expand the **File Settings** panel to review or update the batch configuration. The Order Number Start field is read-only and auto-populated — no action is needed.

Step 5 — Review Column Mapping (optional) The **Column Mapping** panel shows how spreadsheet columns are mapped to EDI fields. Auto-mapping occurs on upload; expand the panel to override any mapping manually.

Step 6 — Generate Click **Generate EDI File**. The app groups rows into orders, builds all EDI records, switches to the EDI Preview tab with colour-coded output, and enables the Download button.

Step 7 — Download Click **Download .txt** to save the EDI file using the standard filename convention.

Order Template — Column Reference

The order template contains 16 columns in a fixed order. All column names must be preserved exactly, including any trailing spaces.

#	Column Header	EDI Field	Notes
1	`Order Ref`	Order / item reference	Groups rows into a single order when shared with the same delivery name and company
2	`ISSN`	ISBN / ISSN (D1)	Normalised to 13 digits in the EDI output
3	`Journal/ Issue Title`	Title	Not written to EDI; for reference only (note: two spaces after `/`)
4	`Volume Number`	—	Not written to EDI; for reference only (note: trailing space)
5	`Volume Part`	—	Not written to EDI; for reference only
6	`Year`	—	Not written to EDI; for reference only
7	`Quantity`	D1 quantity	Falls back to Default Qty setting if blank or zero
8	`Delivery Name`	H2 contact name	Note: trailing space

#	Column Header	EDI Field	Notes
9	`Delivery Company name`	H2 city / company	
10	`Delivery address line 1`	H2 address line 1	
11	`Delivery address line 2`	H2 address line 2	
12	`Delivery address line 3`	H2 address line 3	
13	`Delivery Country`	H2 country code	Accepts ISO 2-letter or 3-letter codes; auto-converted to alpha-3
14	`Post code`	H2 post code	
15	`Telephone number`	H2 telephone	Note: trailing space
16	`Email address`	H2 email	

> **Important:** Several column headers contain trailing spaces or non-obvious punctuation (such as the double space in `Journal/ Issue Title`). These are significant — they are part of the exact header string validated on upload. Always use `order_file.xlsx` as your starting point and do not modify the header row.

Row Grouping

Rows that share the same **Order Ref**, **Delivery Company name** and **Delivery Name** are automatically grouped into a single order with multiple D1 line items. Each unique combination produces its own H1 / H2 / H3 block with sequential D1 records beneath it.

Rows with no Order Ref value are each treated as a separate single-line order.

File Settings Reference

The **File Settings** panel (collapsed by default) contains EDI batch configuration. Expand it by clicking the panel header.

Setting	Default	Editable	Description
File ID / Batch Number	`0027816`	Yes	Written into `$$HDR` and `$$EOF` markers. Update per batch if required.
File Prefix	`PO`	No	Used in the output filename only (e.g. `PO.0027816_...`).
Sender Code	`BL00`	No	4-character code written to `$$HDR` and `$$EOF`.
Currency	`GBP`	Yes	3-character ISO currency code written to each H1 record.

Setting	Default	Editable	Description
Payment Terms	FCA	Yes	Incoterms code written to each H3 record. FCA = Free Carrier; DAP = Delivered at Place.
Order Number Start	Auto	No	Timestamp-derived 10-digit seed. Regenerated on each Generate click.
Default Qty	1	Yes	Fallback quantity when the Quantity column is blank or zero.

Order Number Scheme

Order numbers are derived from the date and time at the moment **Generate EDI File** is clicked, ensuring uniqueness across batches without any manual coordination.

Format: 3 + YY + MM + DD + HH + MM (10 digits)

Example: Clicking Generate at 14:23 on 24 February 2026 produces seed 3260224142.

Orders within the same batch increment sequentially from that seed: 3260224142, 3260224143, 3260224144, and so on.

Provided two batches are not generated within the same calendar minute, their order number ranges will never overlap.

Column Mapping Panel

The **Column Mapping** panel (collapsed by default) shows the relationship between each EDI field and the corresponding spreadsheet column. Auto-mapping occurs on upload using exact header matching.

To override any mapping, expand the panel and use the dropdowns. Selecting (none) leaves that EDI field blank (padded with spaces or zeros) in the output.

Changes take effect on the next Generate click.

EDI Output Format

The generated file is a fixed-width ASCII plain-text file with CRLF (\r\n) line endings. Fields occupy exact character positions with no delimiters. Text fields are space-padded on the right; numeric fields are zero-padded on the left.

File structure:

```
$$HDR{sender}  {fileId}   {timestamp}
H1...  (350 chars)
H2...  (358 chars)
H3...  ( 20 chars)
D1...  (266 chars each — one per line item)
```

```
  [H1 / H2 / H3 / D1 blocks repeat for each order]
  $$EOF{sender}  {fileId}   {timestamp}{recordCount}
```

The $$EOF record count is a 7-digit zero-padded total of H1 + H2 + H3 + D1 lines only. $$HDR and $$EOF themselves are not included in the count.

Record Type Reference

$$HDR — File Header

Chars	Content
0–4	$$HDR
5–8	Sender code (BL00)
9–10	Two spaces
11–17	File ID (7 chars, zero-padded)
18–20	Three spaces
21–34	Timestamp YYYYMMDDHHMMSS

H1 — Order Header (350 characters)

Position	Length	Content
0	2	H1
2	15	Order number
17	8	Order date YYYYMMDD
25	30	Spaces
55	1	C (currency flag)
56	1	Space
57	28	Customer / order reference
85	7	Spaces
92	8	Carrier code — RMA (Royal Mail)
100	2	N
102	30	Spaces
132	57	Zeros
189	6	Spaces
195	40	PDF placeholder (.PDF padded to 40 chars)

Position	Length	Content
235	2	Spaces
237	3	Currency code (e.g. GBP)
240	110	Spaces

H2 — Customer / Address (358 characters)

Position	Length	Content
0	2	H2
2	15	Order number
17	27	Customer code (ST + subscription ref)
44	50	Contact name
94	50	Address line 1
144	50	Address line 2
194	50	Address line 3
244	50	Email address
294	32	City / company name
326	9	Post code
335	3	Country code (ISO alpha-3)
338	20	Telephone number

H3 — Payment Terms (20 characters)

Position	Length	Content
0	2	H3
2	15	Order number
17	3	Incoterms code (FCA or DAP)

D1 — Line Item (266 characters)

Position	Length	Content
0	2	D1
2	15	Order number
17	25	Item reference (subscription number)

Position	Length	Content
42	5	Line number zero-padded (`00001`)
47	3	Spaces
50	18	Zeros
68	78	Spaces
146	28	Quantity block (`0000001` + 21 zeros)
174	12	Unit price in pence (always zero)
186	40	Spaces
226	13	ISBN / ISSN (13 digits)
239	27	Spaces

$$EOF — File Footer

Chars	Content
0–4	`$$EOF`
5–8	Sender code
9–10	Two spaces
11–17	File ID
18–20	Three spaces
21–34	Timestamp
35–41	Record count (7 digits, zero-padded) — H1 + H2 + H3 + D1 only

Output Filename Convention

```
{prefix}.{fileId}_{HHMM}_({DD-MM-YY}).txt
```

Example: `PO.0027816_1423_(24-02-26).txt`

XML Metadata Generator

How to Use

Step 1 — Download the template (first time only) Click **Download Metadata Template** in the Generate XML Metadata panel to save `metadata_template.xlsx`.

Step 2 — Populate the template Fill in one row per title. The template contains three columns: `ISSN`, `Title`, and `Page Extent`. All other specification values are calculated automatically.

Step 3 — Upload the file Drag the populated file onto the metadata upload zone or click to browse. The app validates the column structure and checks every ISSN value. If either check fails, the file is rejected with a detailed error report.

Step 4 — Generate Click **Generate XML & Download ZIP**. Two files are downloaded:

- `metadata.zip` — contains one `.xml` file per row, each named by its ISSN
- `metadata_summary.txt` — a plain-text summary report of all generated records

Step 5 — Clear (optional) Click **Clear** to reset the metadata panel and upload a new file.

Metadata Template — Column Reference

The metadata template contains exactly three columns. Column names and order must not be changed.

#	Column Header	Description
1	`ISSN`	13-digit ISSN with no spaces or hyphens. Used as the XML filename (`{ISSN}.xml`).
2	`Title`	Full journal or issue title. Written directly to `<title>` in the XML.
3	`Page Extent`	Total number of pages. Drives the paper type selection and spine size calculation.

ISSN Validation

After column structure validation passes, every ISSN value in the file is checked individually. An ISSN is valid only if it:

- Contains exactly **13 digits**
- Contains **no spaces**
- Contains **no hyphens**
- Contains **no other non-numeric characters**

If any ISSNs fail this check, the entire file is rejected and a row-by-row error report is shown listing the row number and the invalid value. The file must be corrected before generation can proceed.

Derived Values

Only ISSN, Title and Page Extent are read from the spreadsheet. All remaining specification values are fixed constants or calculated from Page Extent:

XML Field	Source	Value / Rule
`<trim_height>`	Fixed	245

XML Field	Source	Value / Rule
`<trim_width>`	Fixed	`170`
`<binding_style>`	Fixed	`Limp`
`<lamination>`	Fixed	`Matt`
`<paper_type>`	Derived from Page Extent	`Magno Matt 130 gsm` if extent ≤ 32; `Magno Matt 90 gsm` if extent ≥ 33
`<spine_size>`	Calculated	See formula below
`<page_extent>`	From spreadsheet	Passed through directly

Spine Size Calculation

The spine size is calculated using the standard book spine formula, with a Limp binding addition applied:

```
spine = round( (pageExtent × gsm × volume) / 20000 + 0.65 )
```

Where:

- `gsm` is the paper grammage: `130` (for extent ≤ 32) or `90` (for extent ≥ 33)
- `volume` is `10` for both paper types
- `0.65` is the standard addition for Limp binding
- The result is rounded to the **nearest whole number** (millimetres)

Examples:

Page Extent	Paper Selected	Spine Calculation	Spine Size
16	Magno Matt 130 gsm	(16 × 130 × 10) / 20000 + 0.65	2 mm
32	Magno Matt 130 gsm	(32 × 130 × 10) / 20000 + 0.65	3 mm
33	Magno Matt 90 gsm	(33 × 90 × 10) / 20000 + 0.65	2 mm
120	Magno Matt 90 gsm	(120 × 90 × 10) / 20000 + 0.65	6 mm

XML Output Structure

Each generated XML file follows this structure:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<book>
    <basic_info>
        <issn>9771472645051</issn>
        <title>[2026] 1 FCR 4</title>
    </basic_info>
```

```xml
    <specifications>
        <dimensions>
            <trim_height>245</trim_height>
            <trim_width>170</trim_width>
            <spine_size>6</spine_size>
        </dimensions>
        <materials>
            <paper_type>Magno Matt 90 gsm</paper_type>
            <binding_style>Limp</binding_style>
            <lamination>Matt</lamination>
        </materials>
        <page_extent>120</page_extent>
    </specifications>
</book>
```

Files are named `{ISSN}.xml` and bundled into `metadata.zip`.

Summary Report

A `metadata_summary.txt` file is downloaded separately alongside the ZIP. It contains a dated header, one entry per generated XML file, and a footer with totals. Example:

```
========================================
  BLOOMSBURY PUBLISHING — XML METADATA GENERATION SUMMARY
========================================

  Generated : 28 February 2026 at 09:45:12
  Source    : 3 rows processed
_____


    1. ISSN        : 9771472645051
       Title       : [2026] 1 FCR 4
       Page Extent : 120
       Paper Type  : Magno Matt 90 gsm
       Spine Size  : 6 mm
       Trim Size   : 245 × 170 mm
       Binding     : Limp  |  Lamination: Matt
       File        : 9771472645051.xml

_____

  Total XML files generated : 3
========================================
```

Column Validation

Both upload zones enforce strict column validation before any data is processed. The validation checks:

1. **Column count** — the file must have exactly the expected number of columns

2. **Column names** — each header must match the template exactly, including case, spacing, and punctuation
3. **Column order** — columns must appear in the same sequence as the template

If any check fails, the file is rejected immediately and a structured error panel is shown. Each mismatch is listed individually with its column number, the expected header, and what was actually found. No data is loaded until all errors are resolved.

For metadata uploads, a further **ISSN validation** step runs after the column check passes (see ISSN Validation).

The **Download Order Template** and **Download Metadata Template** buttons in each panel provide the correct template files to use as a starting point.

Template Downloads

Both template files must be present in the application's root directory (alongside `index.html`) for the download buttons to work.

Button	File	Panel
Download Order Template	`order_file.xlsx`	Import Excel Data
Download Metadata Template	`metadata_template.xlsx`	Generate XML Metadata

These files define the canonical column structure enforced by the upload validation. The header rows must not be modified.

Dependencies

All libraries are loaded from CDN and require an internet connection on first load. The browser will cache them for subsequent offline use.

Library	Version	Purpose	Source
SheetJS (xlsx)	0.18.5	Excel and CSV parsing	cdnjs.cloudflare.com
JSZip	3.10.1	ZIP file creation for XML bundle	cdnjs.cloudflare.com
Font Awesome	6.5.1	UI icons	cdnjs.cloudflare.com
IBM Plex Sans + Mono	—	Interface and monospace typography	fonts.googleapis.com

Browser Compatibility

Browser	Support
Chrome 90+	✅ Full support
Edge 90+	✅ Full support

Browser	Support
Firefox 88+	✅ Full support
Safari 14+	✅ Full support
Internet Explorer	❌ Not supported

The Clipboard API requires HTTPS or `localhost`. On plain HTTP, copy-to-clipboard will silently fail — use the Download button instead.

Development Notes

EDI Field Positions

All character positions in `script.js` were verified by byte-level comparison against live production files:

- `T1.M02221600__22-02-26_.txt` (TFUK reference)
- `PO_TJ_20260223-190749-830.txt` (CUP reference)

Where the `hachette_order_file_format_spec.md` specification document disagreed with the actual files, the implementation follows the **actual files**.

Adding a New EDI Column

1. Add the exact header string to `ORDER_TEMPLATE_COLUMNS` in `script.js` at the correct position.
2. Add an entry to `EDI_FIELDS` with a unique `key`, display `label`, and `default` header name.
3. Add a `getCell(row, 'yourKey')` call at the appropriate character position inside `generateEDI()`.
4. Update this README.

Changing Fixed XML Metadata Values

The fixed values (trim height, trim width, binding style, lamination) are defined as constants inside `buildXML()` in `script.js`. Locate the `Fixed values` comment block and update the relevant string.

Changing the Paper / Extent Threshold

The extent threshold (currently ≤ 32 = Magno Matt 130 gsm, ≥ 33 = Magno Matt 90 gsm) is controlled by a single `if (extent <= 32)` condition inside `buildXML()`. The paper names, grammage values and volume constant (`10`) are all defined in the same function and can be updated independently.

Changing the Spine Formula

The spine calculation uses the formula `(extent × gsm × volume) / SPINE_FACTOR + LIMP_ADDITION` with `Math.round()`. The constants `SPINE_FACTOR` (20000), `LIMP_ADDITION` (0.65) and `VOLUME` (10) are declared as local constants inside `buildXML()` and can be adjusted there.

Changing the Carrier Code

The Royal Mail carrier code at H1 positions `[92:100]` is hardcoded as `'RMA     '` (8 characters). To change it, locate this line in `generateEDI()` and update the string, ensuring it remains exactly 8 characters:

```
+ 'RMA     '   // [92:100] Royal Mail carrier code
```