Amazon UPC Scraping System (4,000–5,000 UPCs per Day)

---

1. Document Purpose

This document defines the **low-level technical architecture** for the Amazon UPC scraping system. It explains, step by step, **how data flows from Excel input to CSV output**, including all internal components, data structures, and processing logic.

This document is intended for:

- Backend developers
- System implementers
- Internal technical review

---

2. Scope & Constraints

In Scope

- Excel (.xlsx) input
- Amazon public page scraping (search + product pages)
- Brand & UPC validation
- CSV export
- Background job execution

Out of Scope

- Amazon APIs
- Paid scraping APIs
- Proxies or IP rotation
- Parallel scraping
- Browser automation (Puppeteer)

---

3. Target Throughput

- **Daily target:** 4,000–5,000 UPCs
- **Execution mode:** Sequential, background worker
- **Requests per UPC:** ~3 (1 search + 1–2 product pages)
- **Average delay:** 6–10 seconds between requests

---

4. High-Level Component Overview

Excel Upload
↓
API Layer
↓
Job Manager
↓
Scraper Worker
↓
Result Storage
↓
CSV Export

Each component is independent and restart-safe.

---

5. Component-Level Design

5.1 Excel Upload & Parsing Module

Responsibility

- Accept Excel file
- Validate required columns
- Normalize rows into internal format

Input

- `.xlsx` file

Required Columns

Column	Mandatory
UPC	Yes
Brand	Yes

Process Flow

1. User uploads file via UI
2. API receives file using Multer

3. File saved to disk
4. Excel parsed using `xlsx`
5. Rows converted to normalized JSON

Internal Data Structure

```
{
  "rowId": 1,
  "upc": "720476124771",
  "brand": "Ergodyne",
  "status": "PENDING"
}
```

5.2 Job Management Module

Responsibility

- Track progress
- Maintain job state
- Enable resume/restart

Job Metadata Structure

```
{
  "jobId": "JOB_001",
  "status": "QUEUED",
  "totalUpcs": 5000,
  "processed": 0,
  "failed": 0,
  "startedAt": null,
  "completedAt": null
}
```

Job States

```
QUEUED → RUNNING → COMPLETED
          ↓
        PAUSED / FAILED
```

5.3 Sequential Job Queue

Responsibility

- Ensure only one UPC is processed at a time

- Prevent burst traffic

UPC Lifecycle

PENDING → IN_PROGRESS → DONE | FAILED

Worker always fetches the **next PENDING UPC**.

5.4 Scraper Worker (Core Engine)

This module performs all Amazon interactions.

5.4.1 Amazon Search Page Scraping

Input

- UPC code

Request URL

https://www.amazon.com/s?k=<UPC>

Rules

- Public page only
- No cookies
- No login
- Browser-like headers

Output

```
[
  {
    "asin": "B08XXXX",
    "url": "https://www.amazon.com/dp/B08XXXX"
  }
]
```

5.4.2 Sponsored Result Filtering

- Ignore sponsored blocks
- Select only organic results

- Limit to **maximum 3 ASINs**

5.4.3 UPC Fallback Logic

If no result found:

1. Prepend `00` to UPC
2. Retry search once
3. If still empty → mark UPC as `NOT_FOUND`

5.5 Product Detail Scraping Module

Input

- ASIN

Request URL

https://www.amazon.com/dp/<ASIN>

Fields Extracted

Field	Source
ASIN	URL
Brand	“Visit the X Store”
Rating	Star rating
Review Count	Global ratings
UPC	Item Details
BSR	Best Seller Rank
Competitors	Offer listing

Parsed Output


```
{
  "asin": "B08XXXX",
  "brand": "Ergodyne ABC",
  "rating": 4.5,
  "reviews": 12644,
```

```
  "upc": "720476124771",
  "bsr": 193008
}
```

5.6 Validation Engine

Brand Match Logic

scrapedBrand.includes(inputBrand)

UPC Match Logic

scrapedUPC.includes(inputUPC)

Output Flags

```
{
  "brandMatch": true,
  "upcMatch": true
}
```

5.7 Rate Limiting & Throttling

Purpose

- Prevent blocking
- Mimic human behavior

Delay Rules

Step	Delay
Search → Product	5–7 sec
Product → Product	5–7 sec
UPC → UPC	6–10 sec

Implementation

await delay(random(6000, 10000))

5.8 Result Persistence Layer

Storage

- JSON file per job

Example Entry


```
{
  "rowId": 1,
  "inputUpc": "720476124771",
  "inputBrand": "Ergodyne",
  "results": [
    {
      "asin": "B08XXXX",
      "brand": "Ergodyne ABC",
      "brandMatch": true,
      "upcMatch": true,
      "rating": 4.5,
      "reviews": 12644,
      "bsr": 193008
    }
  ],
  "status": "DONE"
}
```

5.9 CSV Export Module

Trigger

- Job reaches COMPLETED state

CSV Structure

Input UPC
Input Brand
ASIN
Amazon Brand
Brand Match
UPC Match
Rating
Reviews
BSR
Status

Output Path

/storage/exports/JOB_001.csv

---

6. End-to-End Data Flow Summary

Excel Upload
↓
Excel Parsing
↓
Job Creation
↓
UPC Queue
↓
Amazon Search
↓
Product Page
↓
Validation
↓
Result Storage
↓
CSV Export

---

7. Failure Handling & Recovery

Scenario	Action
HTTP 503 / CAPTCHA	Pause 30–60 min
Single UPC failure	Mark FAILED
Worker crash	Resume from last UPC
Partial data	Save available fields

---

8. Definition of Done

The system is complete when:

- 4K–5K UPCs/day is achievable
- No paid tools are used

- CSV output matches schema
- Jobs resume safely after failure
- Scraper runs continuously without blocking
