Ingestion Service – Postman Step-by-Step Testing Guide

A detailed, click-by-click guide to test the Ingestion Service using Postman.

Prerequisites

- **Postman** installed
- **MongoDB** running on `localhost:27017`
- **Ingestion Service** running on `http://localhost:8082`
- **Test file**: `ingestion-service/Datasets/Auto_Insurance.csv` (or `Health_Insurance.csv`, `Life_Insurance.csv`)

Optional: Collection Variables

1. Click **Collections** in the left sidebar (or create a new collection).
2. Right-click your collection → **Edit**.
3. Open the **Variables** tab.
4. Add:

Variable	Initial Value	Current Value
`baseUrl`	`http://localhost:8082`	`http://localhost:8082`
`jobId`	*(leave empty)*	*(leave empty)*

5. Click **Save**.

Use `{{baseUrl}}` and `{{jobId}}` in URLs below.

Step 1: Upload File (Phase 3)

1.1 Create a new request

1. Click **New** → **HTTP Request** (or press Ctrl+N).

2. Name it: `1. Upload File`.
3. Save to your collection.

1.2 Configure the request

Field	Value
Method	POST
URL	`http://localhost:8082/api/v1/ingestion/upload`
(or with variable)	`{{baseUrl}}/api/v1/ingestion/upload`

1.3 Set Body

1. Click the **Body** tab.
2. Select **form-data** (not x-www-form-urlencoded, not raw).
3. Add rows:

KEY	TYPE	VALUE
`file`	File	Click **Select Files** and choose `Auto_Insurance.csv` (or any .xlsx/.csv from `Datasets/`)
`insurerId`	Text	`HDFC_LIFE`
`uploadedBy`	Text	`test-user`

> **Note:** Ensure `file` is set to **File** (dropdown next to the key), not Text.

1.4 Send and verify

1. Click **Send**.
2. **Expected:** Status `201 Created`.
3. **Expected Body:**

```
{
  "jobId": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
  "status": "UPLOADED"
}
```

1.5 Save jobId

1. **Copy** the `jobId` value from the response.
2. Either:
   - Paste it into the `jobId` collection variable, or
   - Paste it into the next request when asked.

Or use Tests script (auto-save):

1. Open the **Tests** tab of this request.
2. Paste:

```
if (pm.response.code === 201) {
  var json = pm.response.json();
  pm.collectionVariables.set("jobId", json.jobId);
}
```

3. Save. After Send, `{{jobId}}` will be set automatically.

Step 2: Get Job Status (Phase 4 / Phase 7)

2.1 Create request

1. **New** → **HTTP Request**.
2. Name: `2. Get Job Status`.

2.2 Configure

Field	Value
Method	GET
URL	`http://localhost:8082/api/v1/ingestion/status/{{jobId}}`

If not using variables: Replace `{{jobId}}` with the actual UUID from Step 1.

2.3 Send and verify

1. Click **Send**.
2. **Expected:** Status `200 OK`.
3. **Expected Body** (example):

```
{
  "jobId": "...",
  "status": "UPLOADED",
  "processedRecords": 0,
  "totalRecords": 0,
  "filePath": "C:\\...\\storage\\ingestion\\....csv",
  "insurerId": "HDFC_LIFE",
  "createdAt": "2026-02-12T...",
  "updatedAt": "2026-02-12T..."
}
```


4. Check: `status` is `UPLOADED`, `filePath` exists, `insurerId` matches.

Step 3: Update Status to PROCESSING (Phase 4)

3.1 Create request

1. **New** → **HTTP Request**.
2. Name: `3. PATCH Status -> PROCESSING`.

3.2 Configure

Field	Value
Method	PATCH
URL	`http://localhost:8082/api/v1/ingestion/{{jobId}}/status`

3.3 Set Headers

1. Open **Headers** tab.
2. Add (or rely on Body auto-set):

KEY	VALUE
`Content-Type`	`application/json`

3.4 Set Body

1. Open **Body** tab.
2. Select **raw**.
3. Choose **JSON** from the dropdown (right of raw).

4. Enter:

```
{
  "status": "PROCESSING",
  "failureReason": null
}
```

Or simply: `{"status": "PROCESSING"}`

3.5 Send and verify

1. Click **Send**.
2. **Expected:** Status `204 No Content`.
3. Body will be empty.

Step 4: Update Progress (Phase 4)

4.1 Create request

1. **New → HTTP Request**.
2. Name: `4. PATCH Progress`.

4.2 Configure

Field	Value
Method	PATCH
URL	`http://localhost:8082/api/v1/ingestion/{{jobId}}/progress`

4.3 Set Body

1. **Body → raw → JSON**.
2. Enter:

```
{
  "processedRecordsDelta": 10
}
```

4.4 Send and verify

1. Click **Send**.
2. **Expected:** Status `204 No Content`.

Step 5: Get Status Again (Verify Progress)

5.1 Create request (or duplicate Step 2)

Same as Step 2: **GET** `http://localhost:8082/api/v1/ingestion/status/{{jobId}}`.

5.2 Send and verify

1. Click **Send**.
2. **Expected:** `status` = `"PROCESSING"`, `processedRecords` = `10`.

Step 6: Update Status to COMPLETED (Phase 4)

6.1 Create request

1. **New** → **HTTP Request**.
2. Name: `6. PATCH Status -> COMPLETED`.

6.2 Configure

Field	Value
Method	PATCH
URL	`http://localhost:8082/api/v1/ingestion/{{jobId}}/status`

6.3 Set Body

1. **Body** → **raw** → **JSON**.
2. Enter:

```
{
  "status": "COMPLETED",
  "failureReason": null
}
```

Or: `{"status": "COMPLETED"}`

6.4 Send and verify

1. Click **Send**.
2. **Expected:** Status `204 No Content`.

Step 7: Final Status Check

7.1 Create request (or duplicate Step 2)

GET `http://localhost:8082/api/v1/ingestion/status/{{jobId}}`.

7.2 Send and verify

1. Click **Send**.
2. **Expected:** `status` = `"COMPLETED"`, `processedRecords` = `10`.

Optional: BFF Endpoints (Phase 3 & 7 via BFF)

Requires BFF Service on `http://localhost:8080`.

BFF Upload (Phase 3)

Field	Value
Method	POST
URL	`http://localhost:8080/api/bff/upload`
Body	form-data: `file`, `insurerId`, `uploadedBy`

Same as Step 1, but URL uses port **8080** and path `/api/bff/upload`. Response shape is the same (`jobId`, `status`).

BFF Get Status (Phase 7)

Field	Value
Method	GET
URL	`http://localhost:8080/api/bff/upload/status/{{jobId}}`

Same response structure as Ingestion GET status; BFF proxies to Ingestion.

Optional: Test FAILED Transition

1. **Upload** a new file (Step 1) and save the new `jobId`.
2. **PATCH status** → `PROCESSING` (Step 3).
3. **PATCH status** with:

```
{
  "status": "FAILED",
  "failureReason": "Parse error on row 5"
}
```

4. **GET status** → verify `status` = `"FAILED"`, `failureReason` present.

Quick Reference: Request Summary

#	Method	URL	Body
1	POST	`http://localhost:8082/api/v1/ingestion/upload`	form-data: file, insurerId, uploadedBy
2	GET	`http://localhost:8082/api/v1/ingestion/status/{{jobId}}`	-
3	PATCH	`http://localhost:8082/api/v1/ingestion/{{jobId}}/status`	`{"status":"PROCESSING"}`
4	PATCH	`http://localhost:8082/api/v1/ingestion/{{jobId}}/progress`	`{"processedRecordsDelta":10}`
5	GET	`http://localhost:8082/api/v1/ingestion/status/{{jobId}}`	-
6	PATCH	`http://localhost:8082/api/v1/ingestion/{{jobId}}/status`	`{"status":"COMPLETED"}`
7	GET	`http://localhost:8082/api/v1/ingestion/status/{{jobId}}`	-

Import a Postman Collection (Optional)

You can create a JSON file and import it. Example structure:

```
{
  "info": { "name": "Ingestion Service", "schema":
"https://schema.getpostman.com/json/collection/v2.1.0/collection.json" },
  "variable": [
    { "key": "baseUrl", "value": "http://localhost:8082" },
    { "key": "jobId", "value": "" }
  ],
  "item": [
    {
      "name": "1. Upload",
      "request": {
        "method": "POST",
        "url": "{{baseUrl}}/api/v1/ingestion/upload",
        "body": { "mode": "formdata", "formdata": [
          { "key": "file", "type": "file", "src": "" },
          { "key": "insurerId", "value": "HDFC_LIFE", "type": "text" },
          { "key": "uploadedBy", "value": "test-user", "type": "text" }
        ]}
      }
    },
    {
      "name": "2. Get Status",
      "request": {
        "method": "GET",
        "url": "{{baseUrl}}/api/v1/ingestion/status/{{jobId}}"
      }
    }
  ]
}
```

Save as `.json` and use **Import → Upload Files** in Postman.