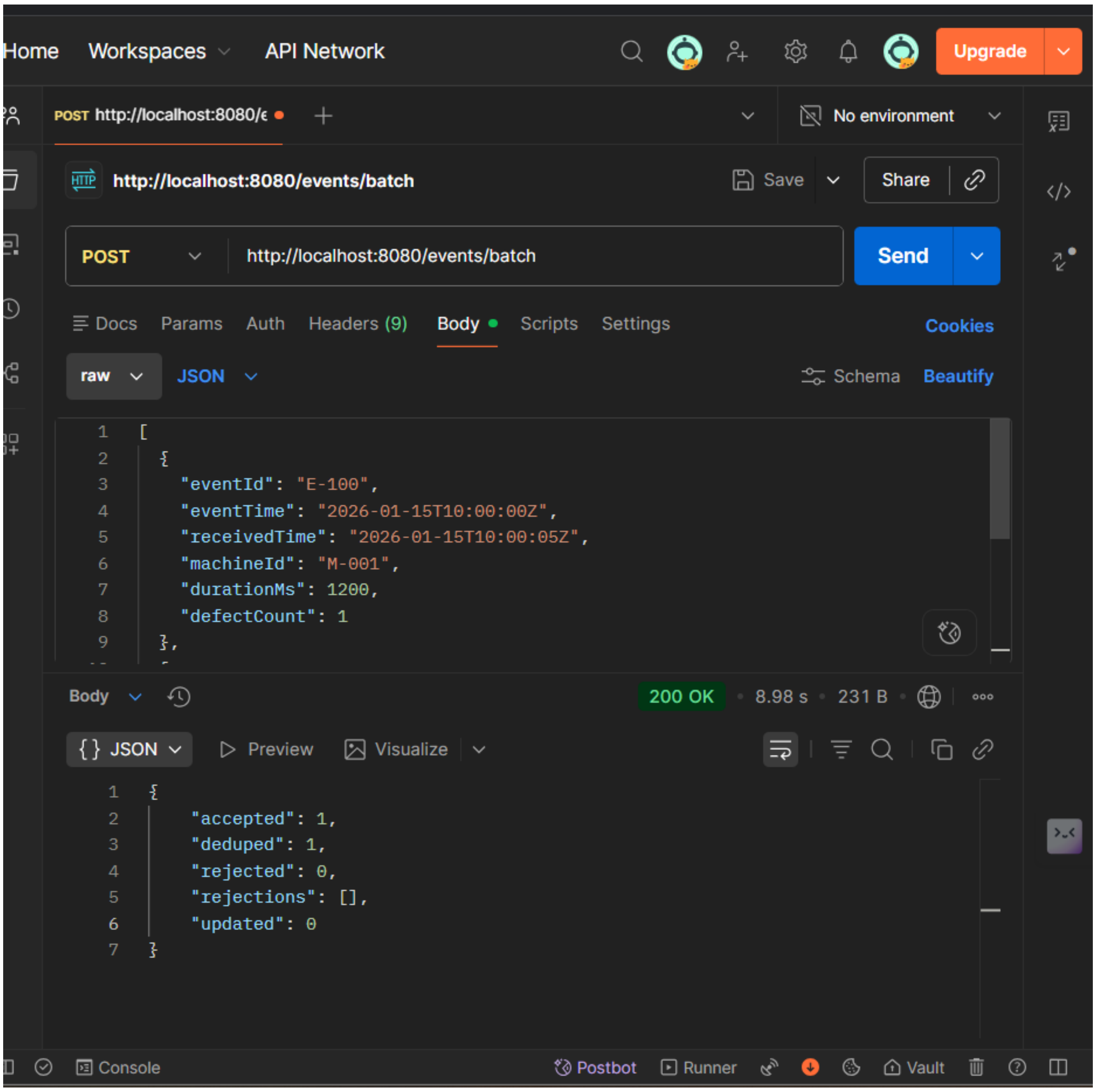
Backend Intern Assignment – Test Case Documentation
Project: Machine Event Ingestion and Statistics System
Tech Stack: Java, Spring Boot
Testing Tool: Postman

Test Case 1: Duplicate Event → Deduplication
Objective: Check that identical events are deduplicated.
Step: Send two events with same eventId and same payload in one batch.
Output:
Accepted = 1, Deduped = 1



POST http://localhost:8080/events/batch

```
[
  {
    "eventId": "E-100",
    "eventTime": "2026-01-15T10:00:00Z",
    "receivedTime": "2026-01-15T10:00:05Z",
    "machineId": "M-001",
    "durationMs": 1200,
    "defectCount": 1
  },
```

200 OK • 8.98 s • 231 B

```
{
    "accepted": 1,
    "deduped": 1,
    "rejected": 0,
    "rejections": [],
    "updated": 0
}
```

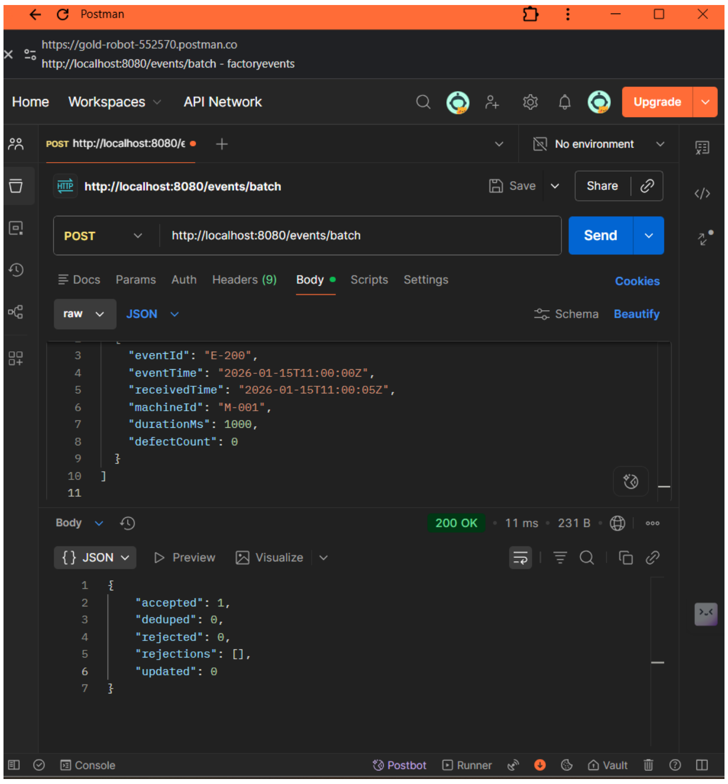
Test Case 2: Same Event ID with New Data → Update

Objective: Check event update when receivedTime is newer.

Step: Send same eventId again with different payload and newer receivedTime.

Output:

Step 1: First request (store original event)



http://localhost:8080/events/batch - factoryevents

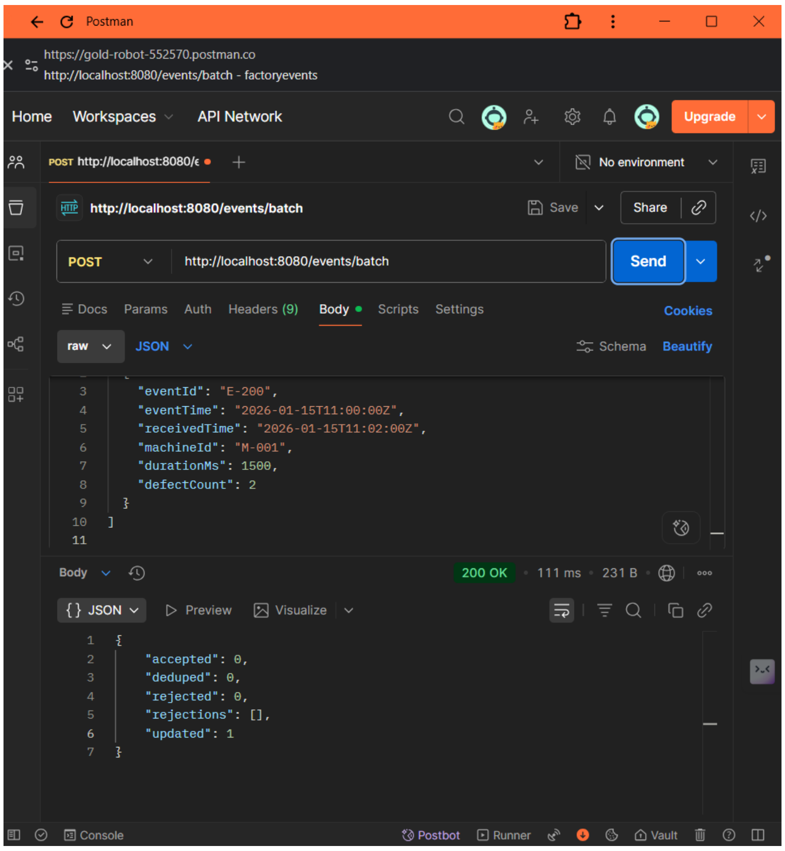
POST http://localhost:8080/events/batch

```
    "eventId": "E-200",
    "eventTime": "2026-01-15T11:00:00Z",
    "receivedTime": "2026-01-15T11:00:05Z",
    "machineId": "M-001",
    "durationMs": 1000,
    "defectCount": 0
  }
]
```

200 OK • 11 ms • 231 B

```
{
  "accepted": 1,
  "deduped": 0,
  "rejected": 0,
  "rejections": [],
  "updated": 0
}
```

Step 2: Second request (same eventId, different payload, newer receivedTime)



http://localhost:8080/events/batch - factoryevents

POST http://localhost:8080/events/batch

```
    "eventId": "E-200",
    "eventTime": "2026-01-15T11:00:00Z",
    "receivedTime": "2026-01-15T11:02:00Z",
    "machineId": "M-001",
    "durationMs": 1500,
    "defectCount": 2
  }
]
```

200 OK • 111 ms • 231 B

```
{
  "accepted": 0,
  "deduped": 0,
  "rejected": 0,
  "rejections": [],
  "updated": 1
}
```

Test Case 3: Older Received Time → Ignore

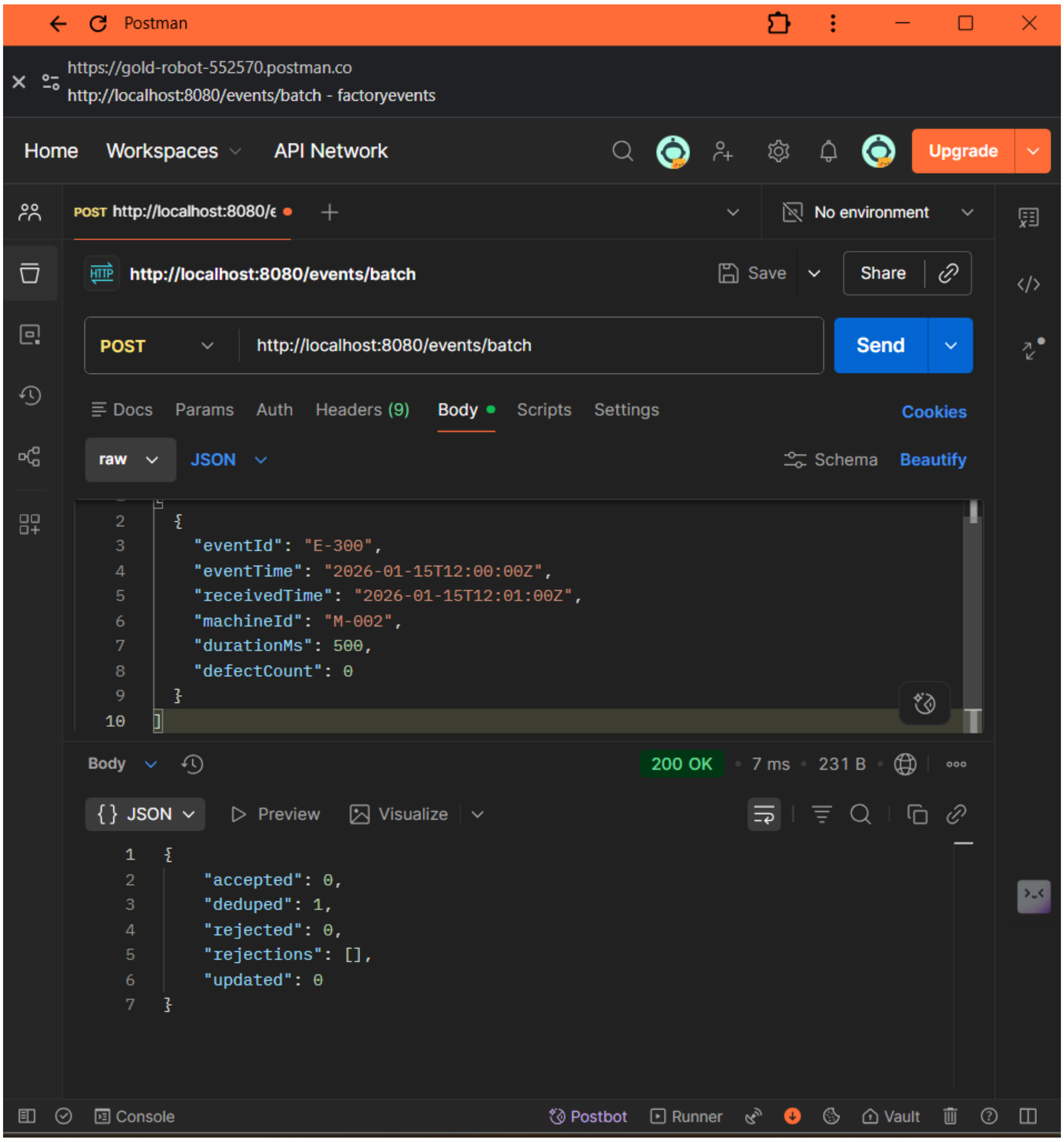
Objective: Check older event does not overwrite newer data.

Step: Send same eventId with older receivedTime.

Output:

Step 1: Insert the newer event first

Step 2: Send an older version of the same event



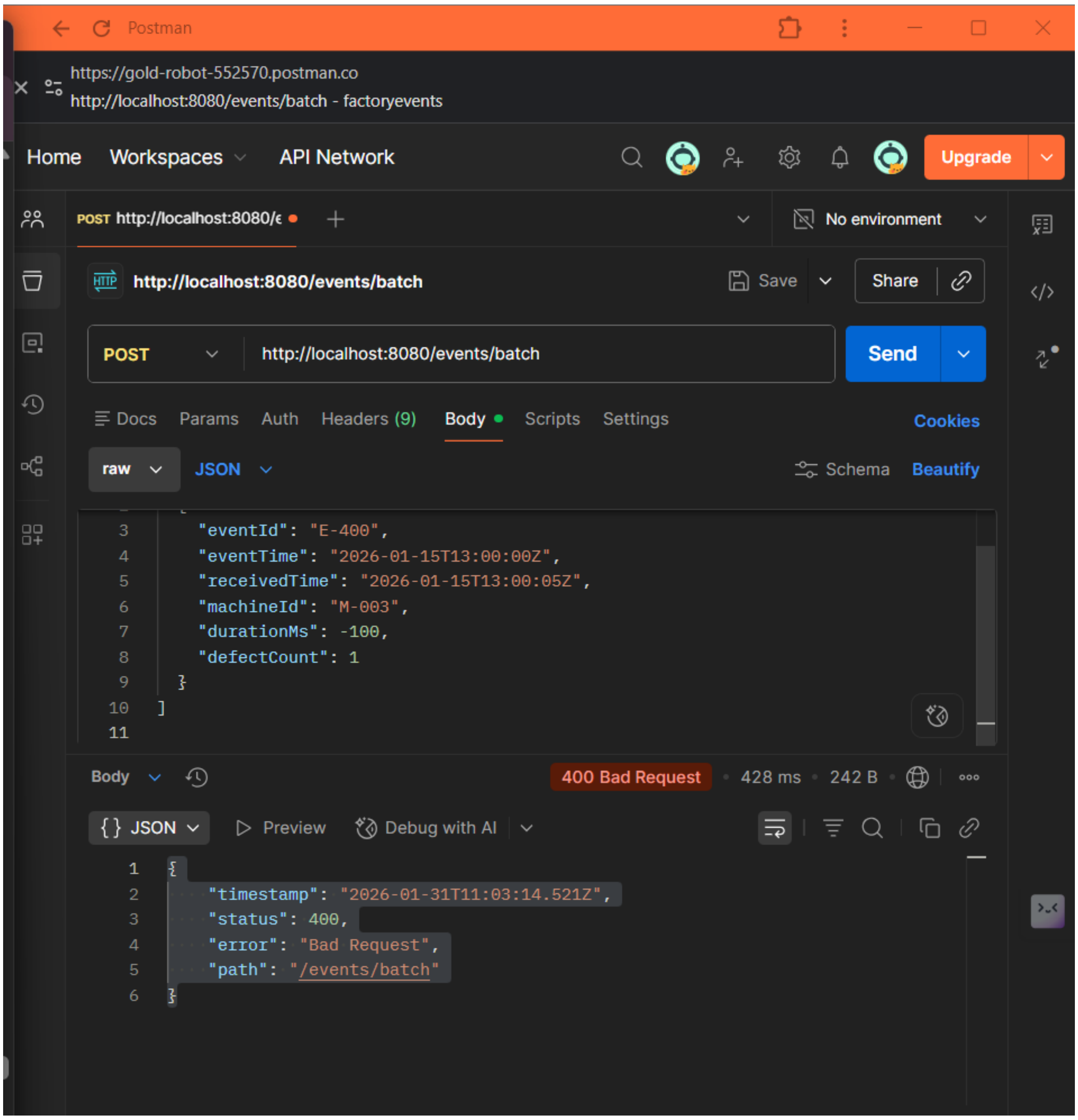
POST http://localhost:8080/events/batch

```
[
  {
    "eventId": "E-300",
    "eventTime": "2026-01-15T12:00:00Z",
    "receivedTime": "2026-01-15T12:01:00Z",
    "machineId": "M-002",
    "durationMs": 500,
    "defectCount": 0
  }
]
```

200 OK • 7 ms • 231 B

```
{
    "accepted": 0,
    "deduped": 1,
    "rejected": 0,
    "rejections": [],
    "updated": 0
}
```

Test Case 4: Invalid Duration → Rejected
Objective: Check rejection for invalid duration.
Step: Send event with negative or very large durationMs.
Output:
Rejected with reason INVALID_DURATION



```
    "eventId": "E-400",
    "eventTime": "2026-01-15T13:00:00Z",
    "receivedTime": "2026-01-15T13:00:05Z",
    "machineId": "M-003",
    "durationMs": -100,
    "defectCount": 1
  }
]
```

```
{
  "timestamp": "2026-01-31T11:03:14.521Z",
  "status": 400,
  "error": "Bad Request",
  "path": "/events/batch"
}
```

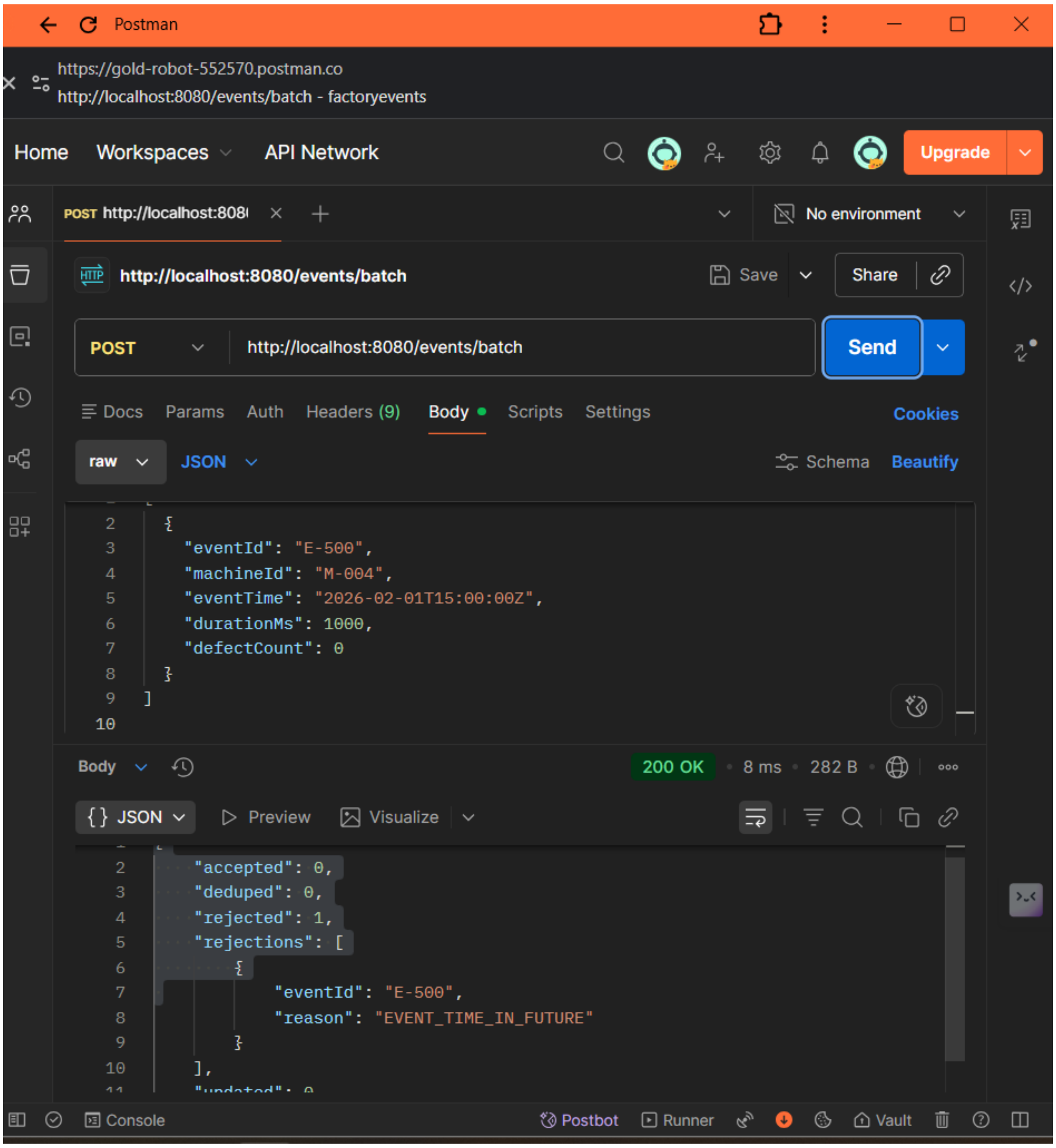
Test Case 5: Future Event Time → Rejected

Objective: Check rejection for future eventTime.

Step: Send event with eventTime more than 15 minutes in future.

Output:

Rejected with reason EVENT_TIME_IN_FUTURE



https://gold-robot-552570.postman.co
http://localhost:8080/events/batch - factoryevents

POST http://localhost:8080/events/batch

```
  {
    "eventId": "E-500",
    "machineId": "M-004",
    "eventTime": "2026-02-01T15:00:00Z",
    "durationMs": 1000,
    "defectCount": 0
  }
]
```

200 OK • 8 ms • 282 B

```
  "accepted": 0,
  "deduped": 0,
  "rejected": 1,
  "rejections": [
      {
          "eventId": "E-500",
          "reason": "EVENT_TIME_IN_FUTURE"
      }
  ],
  "updated": 0
```

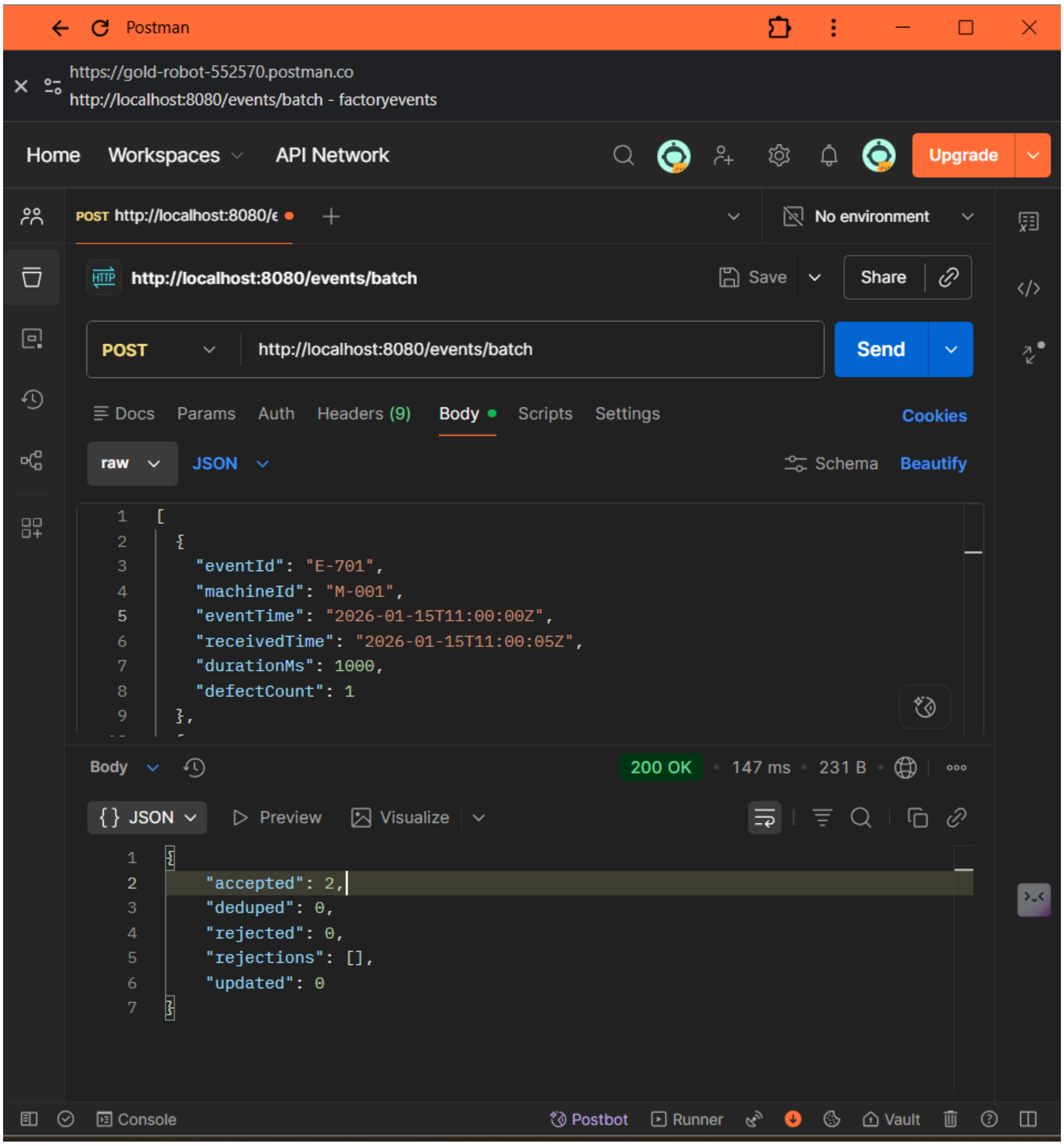
Test Case 6: Valid Single Event → Accepted

Objective: Check normal valid event ingestion.

Step: Send a valid event using /events/batch.

Output:

Accepted = 1



http://localhost:8080/events/batch

```
[
  {
    "eventId": "E-701",
    "machineId": "M-001",
    "eventTime": "2026-01-15T11:00:00Z",
    "receivedTime": "2026-01-15T11:00:05Z",
    "durationMs": 1000,
    "defectCount": 1
  },
```

```
{
  "accepted": 2,
  "deduped": 0,
  "rejected": 0,
  "rejections": [],
  "updated": 0
}
```

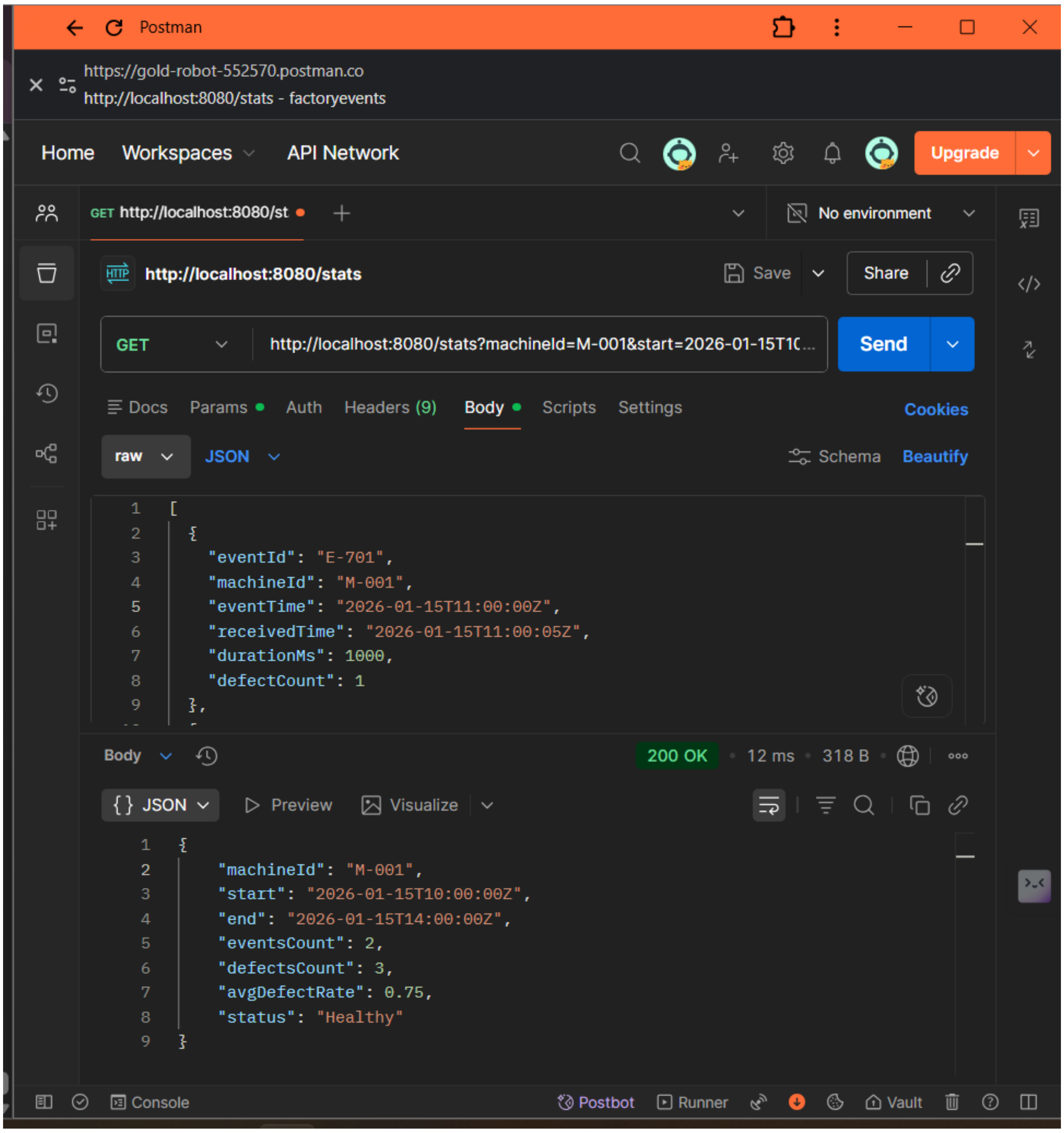
Test Case 7: Stats API with No Events

Objective: Check stats response when no events exist in time range.

Step: Call /stats API for a machine with no events.

Output:

eventsCount = 0, status = Healthy



GET http://localhost:8080/stats?machineId=M-001&start=2026-01-15T1(...

```
[
  {
    "eventId": "E-701",
    "machineId": "M-001",
    "eventTime": "2026-01-15T11:00:00Z",
    "receivedTime": "2026-01-15T11:00:05Z",
    "durationMs": 1000,
    "defectCount": 1
  },
```

```
{
    "machineId": "M-001",
    "start": "2026-01-15T10:00:00Z",
    "end": "2026-01-15T14:00:00Z",
    "eventsCount": 2,
    "defectsCount": 3,
    "avgDefectRate": 0.75,
    "status": "Healthy"
}
```

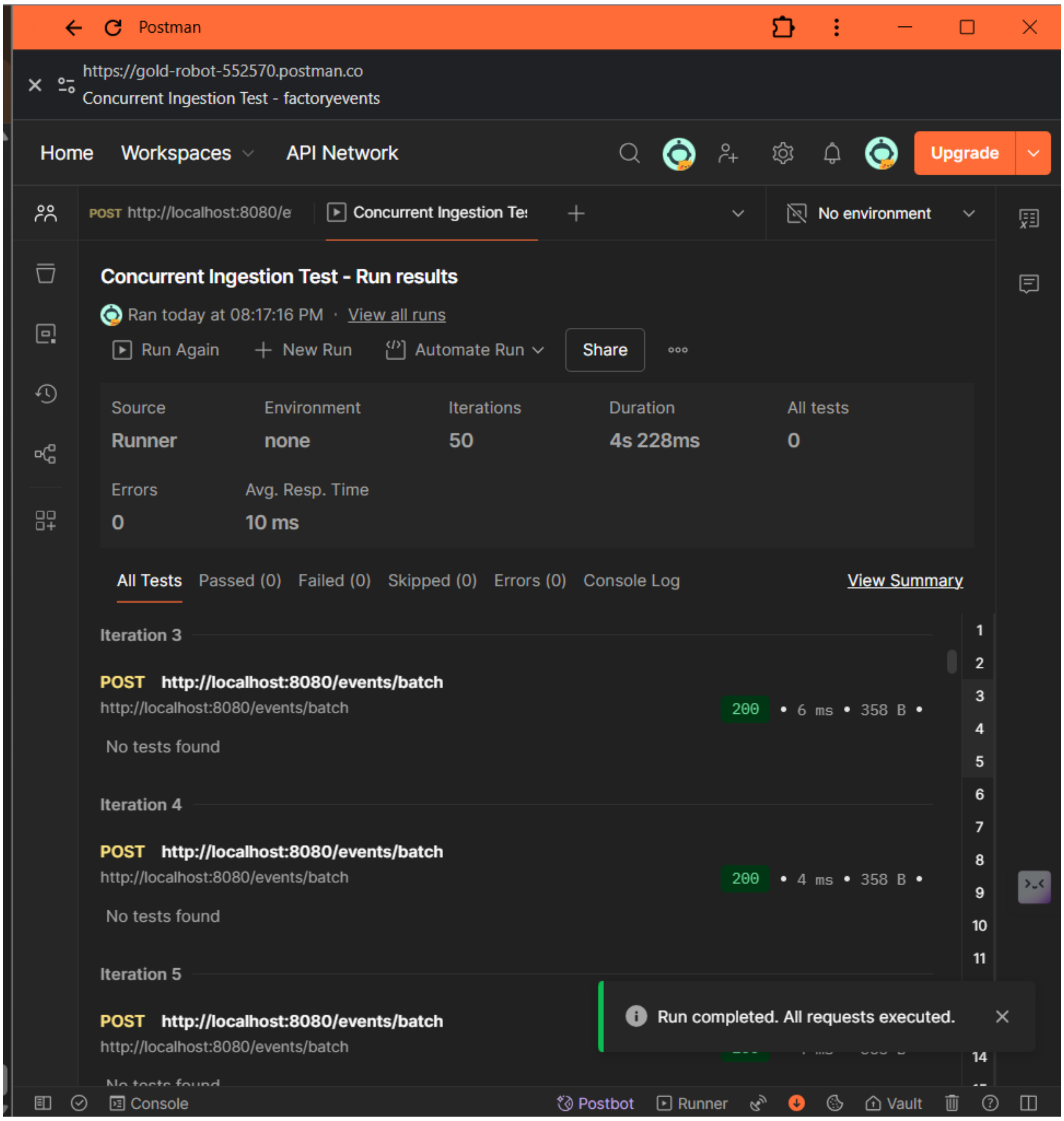
Test Case 8: Concurrent Event Ingestion

Objective: Check system handles multiple requests at same time.

Step: Send multiple batch requests simultaneously with same eventId.

Output:

Only one event stored, others deduped



https://gold-robot-552570.postman.co
Concurrent Ingestion Test - factoryevents

Concurrent Ingestion Test - Run results

Ran today at 08:17:16 PM · View all runs

Source	Environment	Iterations	Duration	All tests
Runner	none	50	4s 228ms	0

Errors	Avg. Resp. Time
0	10 ms

All Tests Passed (0) Failed (0) Skipped (0) Errors (0) Console Log

```
Iteration 3
POST http://localhost:8080/events/batch
http://localhost:8080/events/batch                200 • 6 ms • 358 B •
No tests found

Iteration 4
POST http://localhost:8080/events/batch
http://localhost:8080/events/batch                200 • 4 ms • 358 B •
No tests found

Iteration 5
POST http://localhost:8080/events/batch
http://localhost:8080/events/batch
```

Run completed. All requests executed.