Below is a **brand-new, real-world JIRA-focused implementation** using **Zapier Webhooks** both **as a Trigger** and **as an Action**, and it is **different from all previous prompts** you requested (new Jira use-case, different events, different objectives).

This flow uses **Jira Cloud REST API v3** end-to-end.

✓Jira Cloud + Zapier Webhooks

Trigger + Action (Issue Transitions → Automated Sub-Tasks)

⍰ Business Scenario (Real Use Case)

When a **Jira Issue transitions to "Ready for QA"**, automatically:

□ **Create multiple QA Sub-Tasks**
□ Assign them to the QA team
□ Tag them with automation metadata

This is common in:

- Agile / Scrum teams
- Continuous delivery pipelines
- QA handoff automation

1□□ Webhook as **Trigger**

Jira → Zapier (Issue Transition Event)

Jira Cloud can send **webhooks** for issue updates such as:

- Status changes
- Transitions
- Field updates

We will trigger on:

> **Issue → Transitioned to "Ready for QA"**

A. Create Zapier Trigger

1. Zapier → **Create Zap**

2. Trigger App: **Webhooks by Zapier**
3. Trigger Event: **Catch Hook**
4. Zapier gives you a webhook URL:

```
https://hooks.zapier.com/hooks/catch/556677/jira-ready-qa
```

B. Register Jira Webhook

In **Jira Admin → System → Webhooks → Create Webhook**

Configuration

```
Name: Ready for QA Webhook
URL: https://hooks.zapier.com/hooks/catch/556677/jira-ready-qa
Events: Issue → Updated
```

JQL Filter

```
status CHANGED TO "Ready for QA"
```

C. Payload Sent by Jira

Jira sends JSON similar to:

```
{
  "issue": {
    "id": "10091",
    "key": "PROJ-248",
    "fields": {
      "summary": "Implement payment retry logic",
      "status": {
        "name": "Ready for QA"
      },
       "assignee": {
         "emailAddress": "dev@company.com"
       }
     }
   },
   "user": {
     "displayName": "Dev Lead"
   },
   "timestamp": 1767150900000
 }
```

Zapier automatically parses nested JSON.

D. Zapier Filter Step

Add a **Filter**:

```
Continue only if issue.fields.status.name = "Ready for QA"
```

✅**Trigger complete**

2□□ Webhook as Action

Zapier → Jira API (Create Sub- Tasks)

Now Zapier will create **QA sub- tasks** via Jira REST API.

A. Add Webhooks Action

- App: **Webhooks by Zapier**
- Event: **POST**

B. Jira Cloud Endpoint

```
POST https://your-domain.atlassian.net/rest/api/3/issue
```

C. Headers (Jira Cloud)

```
Authorization: Basic {{JIRA_EMAIL:API_TOKEN_BASE64}}
Content-Type: application/json
Accept: application/json
```

> Jira Cloud uses **Basic Auth with email + API token**.

D. Sub- Task Creation Payload

```
{
  "fields": {
    "project": {
      "key": "PROJ"
    },
    "parent": {
      "key": "PROJ-248"
    },
    "summary": "QA – Regression testing",
    "issuetype": {
      "name": "Sub-task"
    },
    "assignee": {
      "emailAddress": "qa.team@company.com"
    },
    "labels": ["auto-created", "qa"]
  }
}
```

You can repeat this step (or use looping) for:

- Smoke testing
- Regression
- Cross-browser testing

E. Jira Success Response

```
{
  "id": "10142",
  "key": "PROJ-249",
  "self": "https://your-domain.atlassian.net/rest/api/3/issue/10142"
}
```

✓QA sub-task created automatically

□ End-to-End Architecture

```
Developer moves issue → Ready for QA
        ↓
Jira Webhook fires
        ↓
Zapier Catch Hook (Trigger)
        ↓
Filter: Status = Ready for QA
        ↓
Zapier POST → Jira REST API
        ↓
QA Sub-Tasks created automatically
```

✅Why This Is a Real- World Pattern

- Eliminates manual QA task creation
- Enforces consistent testing workflow
- Works without Jira Automation licenses
- Fully audit- tracked in Jira
- Scales across projects

⍰ Security & Reliability

- Jira webhook delivery retries automatically
- Webhooks use HTTPS only
- API token scoped to limited permissions
- Zapier logs all webhook executions

⍰ Optional Enhancements

- Create tasks only for Story / Bug types
- Add priority mapping rules
- Auto- transition issues after all QA tasks closed
- Sync test results from CI tools
- Add Slack or Teams alerts
