webhooks

Webhooks Study Guide

What is a Webhook?

A webhook is:

an automatic event notification sent from one system to another.

Instead of constantly asking:

> “Did something happen?”

the system:

automatically pushes the event.

Simple Explanation

Example:

```
Customer starts chat
      ↓
Glia sends webhook
      ↓
CRM automatically creates ticket
```

The webhook notifies another system:

in real time.

Webhook = Event-Driven Communication

Webhooks are commonly used for:

- notifications
- integrations
- automation
- workflow triggering

Common Webhook Use Cases

Event	Webhook Action
Customer starts chat	Create CRM ticket
Payment completed	Send confirmation
Call ended	Update reporting system
Agent assigned	Notify supervisor
User authenticated	Trigger workflow

How Webhooks Work

Step 1 — Event Occurs

Example:

```
Chat session started
```

Step 2 — Source System Detects Event

Example:

- Glia
- CRM
- cloud platform

Step 3 — Webhook Sent Automatically

Usually:

HTTP POST request

Example:

```
POST /webhook/chat-started
```

Step 4 — Receiving System Processes Event

Example:

```
CRM updates customer interaction history
```

Example Webhook Payload

```
{
  "event": "chat_started",
  "customerId": "12345",
  "timestamp": "2026-05-20T15:30:00Z"
}
```

Webhook Characteristics

Characteristic	Description
Event-driven	Triggered automatically
Push model	Sends data automatically
Real-time	Immediate notifications
Usually HTTP POST	Most common method
JSON payloads	Common data format

Webhooks vs APIs

VERY IMPORTANT INTERVIEW TOPIC

API	Webhook
Request-based	Event-based
Client asks for data	Server pushes data
Pull model	Push model
Requires polling	Real-time notifications
Client initiates communication	Source system initiates communication

---

Simple Analogy

API

Like calling a restaurant:

> “Is my order ready?”

You continuously ask.

Webhook

Like restaurant texting you:

> “Your order is ready.”

Automatic notification.

API Example (Pull Model)

Client requests:

```
GET /api/chat/status
```

Client repeatedly checks status.

Webhook Example (Push Model)

System automatically sends:

```
POST /webhook/chat-completed
```

No polling required.

Why Webhooks Are Important

Webhooks improve:

- automation
- efficiency
- real-time workflows
- integrations
- scalability

VERY common in:

- CCaaS
- SaaS
- banking
- cloud systems

Common Webhook Use Cases In Glia

Likely webhook events:

- chat started
- chat ended
- agent assigned
- authentication completed
- escalation triggered
- interaction transferred

Common Webhook Troubleshooting

Problem 1 — Webhook Not Received

Possible causes:

- incorrect URL
- firewall issue
- endpoint unavailable
- DNS issue

Problem 2 — Authentication Failure

Webhook endpoint may require:

- API key
- bearer token
- OAuth authentication

Possible result:

```
401 Unauthorized
```

Problem 3 — Invalid JSON Payload

Malformed JSON:

```
{
  "event": "chat_started"
  "customerId": "12345"
}
```

Missing comma causes failure.

Problem 4 — Slow Endpoint Response

If receiving system responds slowly:

- webhook timeout
- retries may occur

Problem 5 — SSL/TLS Issues

Webhook endpoints usually require:

HTTPS/TLS

Problems:

- expired certificate
- invalid certificate
- TLS mismatch

Webhook Troubleshooting Flow

Step 1 — Validate Event Triggered

Did source system generate event?

Check:

- logs
- timestamps
- event history

Step 2 — Validate Webhook URL

Check:

- DNS
- endpoint path
- HTTPS
- port access

Step 3 — Validate Authentication

Check:

- bearer token
- API key
- OAuth credentials

Step 4 — Validate Payload

Check:

- JSON syntax
- required fields
- data types

Step 5 — Review HTTP Response Codes

Code	Meaning
200	Success
401	Authentication failed
403	Permission denied
404	Endpoint not found
500	Receiving server failed

Step 6 — Review Logs

Check:

- webhook delivery logs
- API logs
- server logs
- timestamps

Important Webhook Security Concepts

Because webhooks are inbound requests: they should be protected with:

- HTTPS/TLS
- authentication
- validation
- IP restrictions if needed

Common Interview Questions

“What is a webhook?”

Good Answer:

> “A webhook is an event-driven mechanism where one system automatically sends data to another system when a specific event occurs.”

“Difference between API and webhook?”

Good Answer:

> “APIs typically use a request/response pull model where a client requests data, while webhooks use an event-driven push model where the system automatically sends notifications when events occur.”

“Why use webhooks instead of polling APIs?”

Good Answer:

> “Webhooks provide real-time event notifications and reduce unnecessary polling traffic, making integrations more efficient and responsive.”

“How would you troubleshoot webhook

failures?”

Good Answer:

> “I would validate the event trigger, confirm webhook URL connectivity, verify authentication and HTTPS/TLS configuration, review JSON payload formatting, analyze HTTP response codes, and check delivery logs.”

Easy Memory Trick

API = You Ask

Webhook = System Tells You

Example:

```
API → pull
Webhook → push
```

Important Terms To Know

Term	Meaning
API	Request-based communication
Webhook	Event-driven notification

Term	Meaning
Polling	Repeated API checking
Push Model	Automatic event delivery
Pull Model	Client requests data
JSON Payload	Structured webhook data
HTTPS/TLS	Secure webhook transport
Event-Driven	Trigger-based workflow

Revision #1
Created 21 May 2026 00:47:43 by Cesar Gzz
Updated 21 May 2026 00:47:49 by Cesar Gzz