## **🔔 What is a GitHub Webhook?**

A **GitHub Webhook** is a mechanism that allows GitHub to notify Jenkins **instantly** when a specific event occurs in your repository — like a **push**, **pull request**, or **tag**.

## **🔄 Problem With Poll SCM (As per your notes)**

You wrote:

*“Poll SCM: By given time, Jenkins job will trigger* ***with condition****. If no commit → Jenkins skips. If commit → job triggers.”*

✅ True. But this leads to:

*“At 45th min I try to trigger but no commit; commit happens at 46th min → Jenkins must wait till 47th min.”*

This introduces **unnecessary delay**, and can miss real-time responsiveness. That’s where **Webhooks** are a better solution.

## **✅ Why Use GitHub Webhooks?**

*"When commit happens in Git repo, Jenkins job should trigger* ***immediately****."* You also noted:  
 *"Solution: GitHub Hooks"*

✔️ Exactly! GitHub Webhooks remove the delay caused by Poll SCM.  
 ✔️ They let GitHub **push** changes to Jenkins (instead of Jenkins polling GitHub).

## **🛠️ Step-by-Step Setup Using Your Notes + Best Practices**

### **STEP 1: Jenkins Configuration**

Inside your Jenkins Job:

1. Go to **Configure**
2. In **Build Triggers**, check:  
    ✅ **GitHub hook trigger for GITScm polling**

🔍 This tells Jenkins: *"Don’t poll. Just wait for GitHub to notify us."* It uses /github-webhook/ endpoint internally.

### 

### 

### **STEP 2: GitHub Configuration**

You wrote:

*“Goto GitHub → Username → RepoName → Settings → Webhooks → Add webhook”*

Here’s the breakdown:

1. Go to:  
    https://github.com/<your-username>/mahalogin
2. Click **Settings** in the top repo bar.
3. Click **Webhooks** in the left-side menu.
4. Click **Add Webhook**

### **STEP 3: Fill Webhook Details**

*You noted: "Add webhook → Jenkins server URL"*

| **Field** | **Value** |
| --- | --- |
| **Payload URL** | http://<your-jenkins-public-ip>:8080/github-webhook/ |
| **Content type** | application/json |
| **Secret** | *(Optional – can be blank unless you're securing it)* |
| **Events to Trigger** | ✅ *Just the push event* |

💬 Your Note: *“Just the push event”* — correct. You don’t need issue or PR events.

### **STEP 4: Save Webhook**

Click **Add Webhook** on GitHub.  
GitHub will **test the connection** by sending a sample payload to Jenkins.

### **✅ Expected Behavior**

Now, when you push changes to your repo:

* GitHub sends a webhook request to Jenkins.
* Jenkins **immediately triggers the job**.
* No need for polling, no delay.

💡 Your goal from the notes is fulfilled:  
 *"When commit happens, Jenkins job should trigger immediately."*

**💡 Key Differences (Recap)**

| **Feature** | **Poll SCM** | **GitHub Webhook** |
| --- | --- | --- |
| Trigger Style | Jenkins pulls GitHub (polls) | GitHub pushes notification to Jenkins |
| Delay | Possible (based on interval) | Instant trigger |
| Network Load | Higher (frequent polling) | Lower (event-driven) |
| Setup | Only Jenkins | GitHub + Jenkins |

## **🔐 Bonus Tips:**

* Make sure Jenkins is **publicly accessible** for GitHub to reach it (use a tool like [ngrok](https://ngrok.com) during local testing).
* You can view webhook deliveries in GitHub → Settings → Webhooks → **Recent Deliveries**.

Would you like help:

* Getting a **public IP** or using **ngrok** to expose Jenkins locally?
* Adding a **shared secret** for security validation?
* Verifying webhook logs in Jenkins and GitHub?