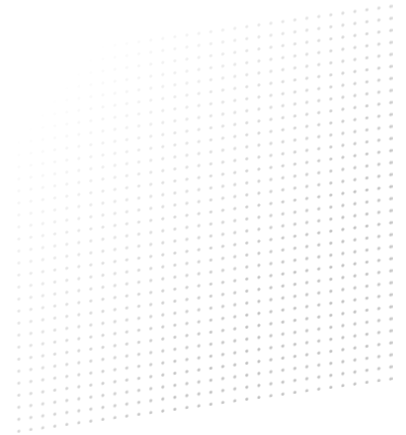

A RESOURCE FOR DESIGN SYSTEM AND DESIGNOPS TEAMS

Component usage Slack bot.



What gets celebrated gets repeated. Celebrate what you can prove.

Scan main. Diff last week. Post the milestones.

22

checks, each with its evidence

04

scripts, no dependencies

00

messages sent while writing this
guide

Make adoption visible where people already are.

The essay behind this guide describes a Slack bot that announced adoption milestones: a team's first component, then 10, 50 and 100. A milestone in the channel makes invisible work visible, to the team that did it and to everyone who has not started. In Field Guide 16's terms, it turns referral and revenue back into awareness.

Such a bot fails in two ways: it posts numbers nobody can check, or it posts something embarrassing, like a regression, a ranking of people or an invented hours-saved figure. This one counts JSX instances on the default branch, explains the count in every message, posts only good news about teams, and does nothing unless given `--post` and a webhook secret.

The kit: four zero-dependency scripts, two fixture repos a week apart, real react-scanner reports of the same code, and a workflow. No message was sent while writing this guide.

Version 1.0 / Sources checked 24 September 2026

Field Guide 18 of the Design × AI series. Pairs with Field Guides 16 (AARRR funnel) and 17 (time-to-first-component benchmark); all three use Acme and its `@acme/ui` library. Verified 24 September 2026: Node 22.22, react-scanner 1.2.0, actions/checkout v7, setup-node v7, upload-artifact v7.

Practical guidance, not a standard. Dry runs only: the posting path was exercised against a stub, never a real webhook, and the workflow was parsed but not executed. Check Slack's and GitHub's current docs before production use. Prepared with AI assistance and edited by hand.

START HERE

Scan, diff, message, post.

Four small steps, each its own script, so you can swap one without touching the others.

Step	Script	In	Out
Scan	<code>scan-usage.mjs</code>	Repo on main, or react-scanner reports	Snapshot: instances per team and component
Diff	<code>diff-usage.mjs</code>	Last snapshot, this snapshot	Milestones, first uses, drops
Message	<code>slack-message.mjs</code>	Diff, config	Block Kit payload, validated
Post	<code>bot.mjs</code>	Payload, <code>SLACK_WEBHOOK_URL</code>	A dry-run print, or one POST
Schedule	<code>ds-celebration.yml</code>	Friday 09:00 UTC	Weekly run, snapshot kept as an artifact

Trying it

Run the bot on the two fixture repos (Appendix A). Nothing leaves your machine.

Wiring it up

Sections 01 and 04, then the workflow in section 05. Record a baseline first.

You already run react-scanner

Feed its reports in with `--react-scanner`. Item B05: never mix sources in one history.

Deciding what to post

Section 02. Teams, milestones and firsts; no drops, no people, no hours saved.

Label	Meaning
<code>DATA</code>	A field in <code>bot.config.json</code> or the snapshot format.
<code>SCRIPT</code>	Enforced or computed by a kit script.
<code>PROTOCOL</code>	A rule about how the bot runs and handles state and secrets.
<code>PRIVACY</code>	Protects people: the bot celebrates teams, not individuals.
<code>LOOP</code>	Feeds adoption back into awareness, the growth loop the essay describes.
<code>PRACTICE</code>	A working method with a review signal, not a hard gate.

SECTION 01

Count what you can prove.

Every number the bot posts must survive someone opening the repo and counting. That means a static scan of the default branch.

Suggested owners: Design system engineering lead

☐ Count JSX instances on main, per team

B01 **SCRIPT** **DATA**

An instance is one rendered element from the system's package. Scan main, so a milestone means shipped. Teams map by path prefix; unmapped code lands in Unassigned, which is never celebrated.

Evidence: Week-39 fixture: 76 instances of 11 components across 5 teams; week 38 had 64. `teams[].paths` in `bot.config.json`.

Kit: `scan-usage.mjs`.

☐ Tests, stories and comments are not usage

B02 **SCRIPT**

react-scanner's default globs skip test and spec files [S12]; the kit also skips stories and commented-out code.

Evidence: The fixture's test file (2 Buttons), story and a comment mentioning `<Button>` add 0.

S12. Kit fixture.

☐ Imports are resolved as written

B03 **SCRIPT**

Aliases count under the real name, namespace usages fold into the component, sub-components keep their path, type-only imports count nothing.

Evidence: `TextField as Field` counts 4 TextField; `<UI.Text>` counts as Text; `<Tabs.List>` as Tabs.List.

Kit fixture.

☐ The regex scanner is checked against an AST scanner

B04 **SCRIPT**

A regex scanner is fast and dependency-free, and wrong on exotic code. Check it against react-scanner on your repo before trusting it.

Evidence: react-scanner 1.2.0 on the week-39 fixture: the same team totals (52, 13, 3, 5, 3) and, with `namespaces: ["UI"]`, the same per-component counts.

S12. Kit: `examples/react-scanner/`.

☐ One source per history

B05 **SCRIPT** **PROTOCOL**

The two scanners name some instances differently (`UI.Text` against `Text`), so diffing across them invents "firsts". Switching source means a new baseline.

Evidence: `diffSnapshots()` refuses snapshots from different sources.

Kit.

If someone can count it by hand and get a different number, do not post it.

SECTION 02

Post good news about teams.

The essay's milestones are a team's first component, then 10, 50 and 100. The bot adds firsts, and leaves out anything that would turn the channel into a scoreboard.

Suggested owners: Design system lead + community lead

☐ The first run records a baseline and posts nothing

B06 **SCRIPT**

With no previous snapshot, everything would be a milestone. Save the snapshot and wait a week.

Evidence: Week-38 run: `baseline: 64 instances of @acme/ui across 5 teams`, no payload.

Kit: `bot.mjs`.

☐ Only the highest threshold crossed is posted

B07 **DATA** **SCRIPT**

Thresholds 1, 10, 50, 100, 250, 500, 1000 on team instances. A team that jumps from 5 to 60 hears about 50, not 10 and 50.

Evidence: Week 39: Checkout 46 to 52 crosses 50; Search 9 to 13 crosses 10.

Essay thresholds, extended. Kit output.

☐ A team's first component is its own message

B08 **SCRIPT** **LOOP**

The first migration is the hardest. It gets its own line, worded as a first, not as a threshold.

Evidence: Week 39: Accounts 0 to 3, posted as "shipped its first `@acme/ui` components".

Essay. Kit output.

☐ Firsts are celebrated: in the organisation, and per team

B09 **SCRIPT** **LOOP**

The first team to ship a component de-risks it for everyone else. A team's first use of an existing component is quieter news, grouped in one block.

Evidence: Week 39: Billing ships the first DatePicker; Search uses Badge and Button for the first time.

Kit output.

☐ No drops, no people, no invented hours

B10 **PRIVACY** **PRACTICE**

Drops go to the run log for a conversation, not the channel. The bot names teams, never individuals. It posts counts it can prove; an hours-saved figure is a model, not a count.

Evidence: Week-39 log: `not posted: Billing went from 6 to 5 instances`. No `@` mentions or time-saved field in the payload builder.

Field Guide 16, items F15 and F20.

Celebrate the team that moved. Talk to the one that slipped.

SECTION 03

Build a message Slack accepts.

Incoming webhooks take JSON with `text` and optional `blocks`. The limits below are from Slack's reference; the kit checks each before anything is sent.

Suggested owners: Design system engineering lead

☐ The webhook owns the channel

B11 `SCRIPT`

An incoming webhook posts to the channel chosen at install; channel, username and icon cannot be overridden [S01]. A `channel` field does nothing, so the validator rejects it.

Evidence: `validatePayload()` flags `channel`, `username`, `icon_emoji` and `icon_url`.
S01.

☐ The top-level text carries the whole news

B12 `SCRIPT`

With blocks, `text` is the notification fallback [S03], and screen readers read it rather than the blocks [S02]. One sentence with every milestone in it.

Evidence: Week-39 payload: a 258-character `text` naming all five pieces of news.
S02, S03.

☐ Block Kit limits are checked before sending

B13 `SCRIPT` `DATA`

At most 50 blocks [S02]; header plain_text up to 150 characters [S05]; section text 1 to 3000, up to 10 fields [S04]; context up to 10 elements [S06]. Busy weeks post the top 20 highlights and "and N more".

Evidence: Week-39 payload: 8 blocks, no problems. A failing payload stops the run. `maxHighlights: 20`.
S02, S04, S05, S06.

☐ Names are escaped

B14 `SCRIPT`

Slack's mrkdwn treats `&`, `<` and `>` as control characters; send them as HTML entities [S07]. A team called `R&D <Labs>` would otherwise break the message.

Evidence: `escapeMrkdwn()` on every team and component name; a unit test with that team name.
S07.

☐ Every message says where its numbers come from

B15 `DATA`

A context line: the commit, the counting method, what is excluded, the org total and a dashboard link.

Evidence: Week 39: scan of main at `9b7e0a3`, tests and stories excluded, 76 instances, up 12.
Kit output.

A message that explains its own numbers does not need a follow-up thread.

SECTION 04

Send only when asked twice.

The bot is harmless until it has a secret and a flag. Keep it that way.

Suggested owners: Design system engineering lead

☐ **Dry run is the default**

B16 **PROTOCOL** **SCRIPT**

Without `--post` the bot prints the payload and sends nothing. Posting needs both the flag and `SLACK_WEBHOOK_URL`.

Evidence: Every run for this guide was a dry run. With `--post` and a non-Slack URL the bot exits 1 before any request.

Kit: `bot.mjs`.

☐ **The webhook URL is a secret**

B17 **PROTOCOL**

Slack warns that the URL contains a secret and revokes leaked ones [S01]. Read it from the environment, never from arguments; never print it; store it as a repository secret.

Evidence: `postToSlack()` accepts only `https://hooks.slack.com/services/` URLs and never logs them. S01.

☐ **Success is HTTP 200 with the body ok**

B18 **SCRIPT**

Incoming webhooks answer 200 and `ok`; errors such as `invalid_payload` or `channel_is_archived` come back as text [S01]. Anything else fails the run.

Evidence: Unit tests against a stub `fetch`: `ok` passes, a 400 `invalid_payload` throws. Never tested against Slack.

S01.

☐ **The snapshot advances only after a successful post**

B19 **PROTOCOL**

If the post fails and the snapshot moves on, that week's news is lost. Save after success, and only from runs that actually post.

Evidence: `--write-state` runs after the post; the workflow uploads only when posting succeeded.

Kit.

The safest bot is one that has to be asked twice before it speaks.

SECTION 05

Run it every Friday.

A weekly rhythm matches the essay's weekly tracking. The workflow's job is to keep the snapshot chain unbroken with the least access possible.

Suggested owners: Design system engineering lead + platform team

☐ Schedule in UTC, and expect delays

B20 **PROTOCOL**

Scheduled workflows use UTC cron on the latest commit of the default branch, can be delayed under load, and are disabled in public repos after 60 days without activity [S08].

Evidence: `cron: "0 9 * * 5"`, plus `workflow_dispatch` for a manual dry run.
S08.

☐ Last week's snapshot is an artifact, not a cache entry

B21 **PROTOCOL**

Cache entries unused for a week are evicted [S09], which a weekly job can hit exactly. The workflow downloads the newest posting run's artifact with `gh run download` [S10, S11].

Evidence: `upload-artifact` with 30-day retention; the download loop picks the newest run that has one.
S09, S10, S11.

☐ The workflow gets the least it needs

B22 **PROTOCOL**

`contents: read` to scan, `actions: read` to fetch the previous artifact, the webhook in secrets. No write access to the repo.

Evidence: `permissions` block in `ds-celebration.yml`.
Kit workflow.

Same day, same scan, same channel. Consistency is what makes a milestone mean something.

APPENDIX A

Week 39, in dry run.

Two fixture repos a week apart: week 38 records the baseline, week 39 produces the payload. Both runs print; neither posts.

terminal

```
$ node scripts/bot.mjs examples/acme-web-week-38 --state state.json --write-state state.json
baseline: 64 instances of @acme/ui across 5 teams

$ node scripts/bot.mjs examples/acme-web-week-39 --state state.json --commit 9b7e0a3 --date
2026-09-25
dry run: 76 instances of @acme/ui across 5 teams
  not posted: Billing went from 6 to 5 instances
  dry run: payload below, nothing sent. Add --post and SLACK_WEBHOOK_URL to send.
```

examples/payload.week-39.json (excerpt)

```
{
  "text": "Design system milestones: Checkout crossed 50 @acme/ui instances; Search crossed 10
@acme/ui instances; Accounts shipped its first @acme/ui components; Billing shipped the first
DatePicker in the organisation; Search used Badge and Button for the first time.",
  "blocks": [
    { "type": "header",
      "text": { "type": "plain_text", "text": "Design system milestones, 2026-09-25", "emoji":
true } },
    { "type": "section",
      "text": { "type": "mrkdwn",
        "text": ":tada: *Checkout* crossed *50* `@acme/ui` instances (now 52, up 6 this
week)." } },
    { "type": "divider" },
    { "type": "section",
      "text": { "type": "mrkdwn", "text": "New to a team this week\n*Search*: first `Badge`,
`Button`" } },
    { "type": "context",
      "elements": [
        { "type": "mrkdwn", "text": "Counts are JSX instances of `@acme/ui` components from a
static scan of main at `9b7e0a3`; tests and stories excluded. Org total: 76 instances, up 12."
},
        { "type": "mrkdwn", "text": "<https://design.acme.example/adoption|Adoption dashboard>"
} ] }
  ]
}
```

Three of the four milestone sections are left out here; the file has all 8 blocks.

APPENDIX B

The weekly workflow.

`workflows/ds-celebration.yml`, copied to `.github/workflows/`. Parsed as YAML by the kit's tests; not executed.

`workflows/ds-celebration.yml` (excerpt)

```
on:
  schedule:
    - cron: "0 9 * * 5" # Fridays 09:00 UTC
  workflow_dispatch:
  inputs:
    post: { description: "Post to Slack (unticked = dry run)", type: boolean, default: false }

permissions:
  contents: read
  actions: read # to download the previous run's snapshot

jobs:
  celebrate:
    runs-on: ubuntu-latest
    env:
      POST: ${ github.event_name == 'schedule' || inputs.post }
    steps:
      - uses: actions/checkout@v7
      - uses: actions/setup-node@v7
        with: { node-version: 22 }
      - name: Download last snapshot # newest posting run's artifact, via gh run download
      - name: Scan, diff and post
        env:
          SLACK_WEBHOOK_URL: ${ secrets.DS_CELEBRATION_WEBHOOK }
        run: |
          node tools/ds-bot/scripts/bot.mjs . --config tools/ds-bot/bot.config.json \
            --state .ds-usage/last.json --write-state .ds-usage/last.json \
            --commit "${GITHUB_SHA::7}" --date "$(date -u +%F)" \
            ${[ "$POST" = "true" ] && echo --post}
      - name: Keep snapshot for next week
        if: success() && env.POST == 'true'
        uses: actions/upload-artifact@v7
        with: { name: ds-usage, path: .ds-usage/last.json, retention-days: 30 }
```

Condensed. The file has the full download step and comments. Check action versions before use.

KEEP WITH THE BOT

Leave a setup record.

One record when the bot goes live, updated whenever its rules change, so the channel knows what the numbers mean.

Channel / webhook owner	Secret name and rotation date
Scan source (built-in or react-scanner)	Scanner checked against AST (date, result)
Team path map reviewed (date)	Milestone thresholds
Baseline run (date, commit)	First posting run (date)
Who follows up on drops	Next review

Change the rules, record a new baseline.

New thresholds, a new team map or a new scanner changes what counts. Start a fresh snapshot rather than diffing across the change.

SOURCES / MAINTENANCE

Keep the guide current.

Sources checked 24 September 2026. Slack moved its developer docs to docs.slack.dev; the limits quoted are from those pages on that date. Action versions were the latest releases on GitHub that day.

S01 / Slack, Sending messages using incoming webhooks

POST JSON; channel, username and icon cannot be overridden; 200 ok; error strings; the URL is a secret.

<https://docs.slack.dev/messaging/sending-messages-using-incoming-webhooks/>

S02 / Slack, Block Kit

Up to 50 blocks per message; screen readers and the top-level text field.

<https://docs.slack.dev/block-kit/>

S03 / Slack, chat.postMessage

With blocks, text is the notification fallback; keep it under 4,000 characters.

<https://docs.slack.dev/reference/methods/chat.postMessage/>

S04 / Slack, Section block

Text 1 to 3000 characters; up to 10 fields of 2000; block_id up to 255.

<https://docs.slack.dev/reference/block-kit/blocks/section-block/>

S05 / Slack, Header block

plain_text only, up to 150 characters.

<https://docs.slack.dev/reference/block-kit/blocks/header-block/>

S06 / Slack, Context block

Up to 10 image or text elements.

<https://docs.slack.dev/reference/block-kit/blocks/context-block/>

S07 / Slack, Formatting message text

Escaping &, < and >; link and bold syntax in mrkdwn.

<https://docs.slack.dev/messaging/formatting-message-text/>

S08 / GitHub Docs, Events that trigger workflows: schedule

UTC cron, default branch, delays under load, 60-day inactivity in public repos.

<https://docs.github.com/en/actions/writing-workflows/choosing-when-your-workflow-runs/events-that-trigger-workflows>

S09 / actions/cache

Caches not accessed within a week are evicted; keys are immutable. v6.1.0.

<https://github.com/actions/cache>

S10 / GitHub Docs, Storing and sharing data from a workflow

Downloading another run's artifacts needs a token and the run id.

<https://docs.github.com/en/actions/writing-workflows/choosing-what-your-workflow-does/storing-and-sharing-data-from-a-workflow>

S11 / GitHub CLI, gh run download

Download a run's artifacts by run id and --name into --dir.

https://cli.github.com/manual/gh_run_download

S12 / react-scanner

AST-based React usage scanner, version 1.2.0: importedFrom, globs and three output processors.

<https://github.com/moroshko/react-scanner>

Maintenance: recheck the Slack incoming-webhook and Block Kit pages and the action versions at each review. Update the PDF, HTML, Markdown and JSON together.