

PICKWARE

# Wer mag Montagmorgen? Langweilige Aufgaben per Chatbot lösen

von Niklas Büchner

PICKWARE

# Eine kurze Einführung in die Slack API

## Langweilige Aufgaben automatisieren

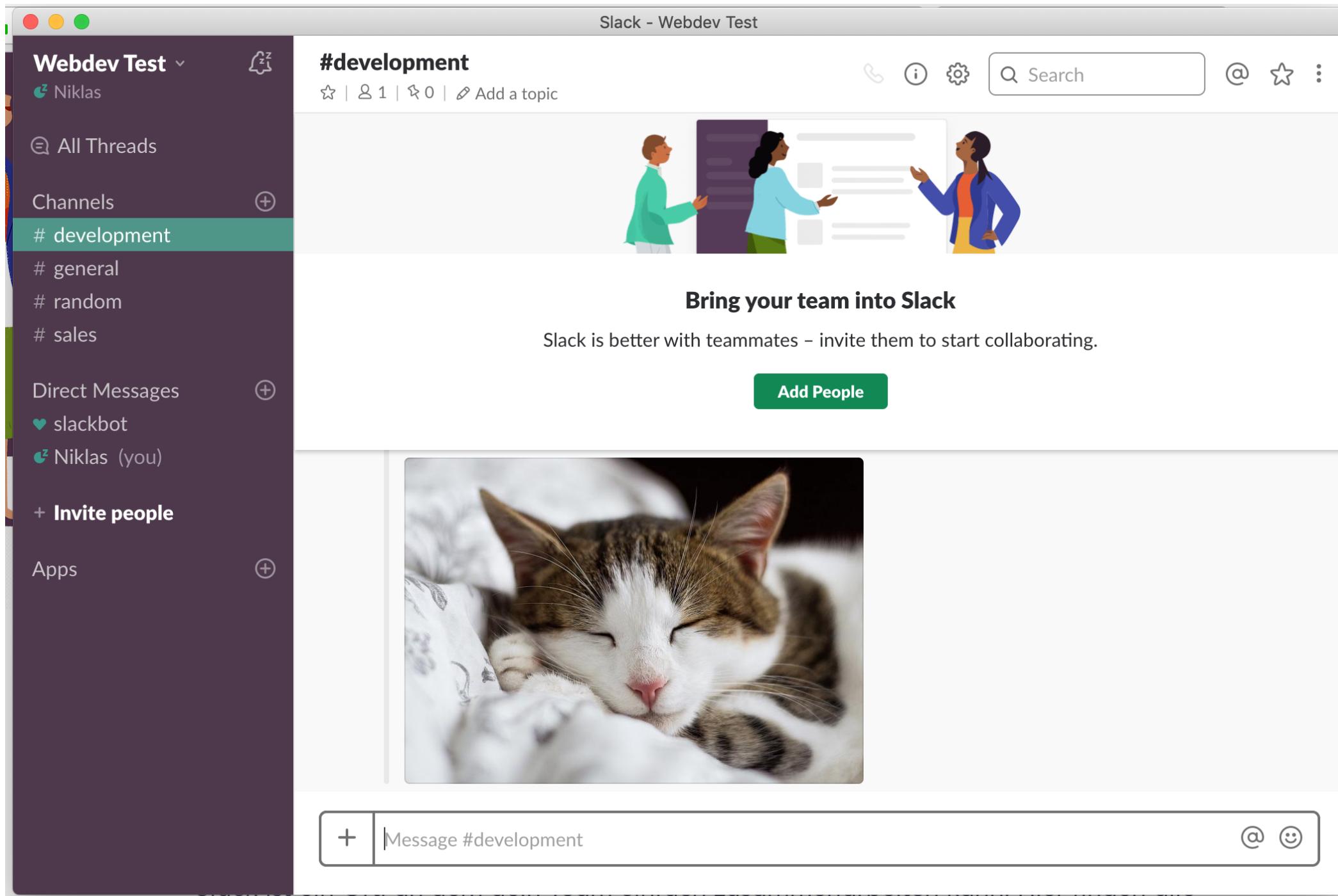
von Niklas Büchner

# Vorstellung

- Niklas
- Softwareentwickler (DevOps)
- E-Commerce
- PHP, JavaScript, (Rust ❤)



# Was ist Slack?



# Die Slack API

```
'https://api.slack.com/apps'
```

## Start here

- Building Slack apps
- Recent updates
- Best practices
- App blueprints

## App features

- Internal integrations
- Incoming webhooks
- Slash commands
- Building bots
- Dialogs
- Shared Channels
- Enterprise Grid
- Legacy: Workspace apps

## Messages

- Introduction
- Basic formatting
- Attaching content

## Your Apps

[Create New App](#)

If your app is (or will be) listed in the Slack App Directory, please review our new [Slack App Directory Agreement](#). These terms are in addition to the existing Developer Policy, API TOS, and Brand Guidelines.

By keeping your app in the App Directory or review process, you're confirming your agreement to the Slack App Directory Agreement and to providing additional information for security review, if requested. If you don't agree with this Agreement, please send an email to [feedback@slack.com](mailto:feedback@slack.com), and we'll remove your app from the App Directory or the review process.

[I Agree](#) Filter apps by name or workspace

### App Name

### Workspace

### Distribution Status



API Introduction

Webdev Test

Not distributed

# Wie lief es gestern?



**API Introduction** APP 11:55 PM

Gestrige Verkaufszahlen

Hardwarebestellungen: **15.720€**

Softwareverkäufe: **15.720€**

# Wie funktioniert der Server?

```
const app = express();
// Weitere Abhängigkeiten

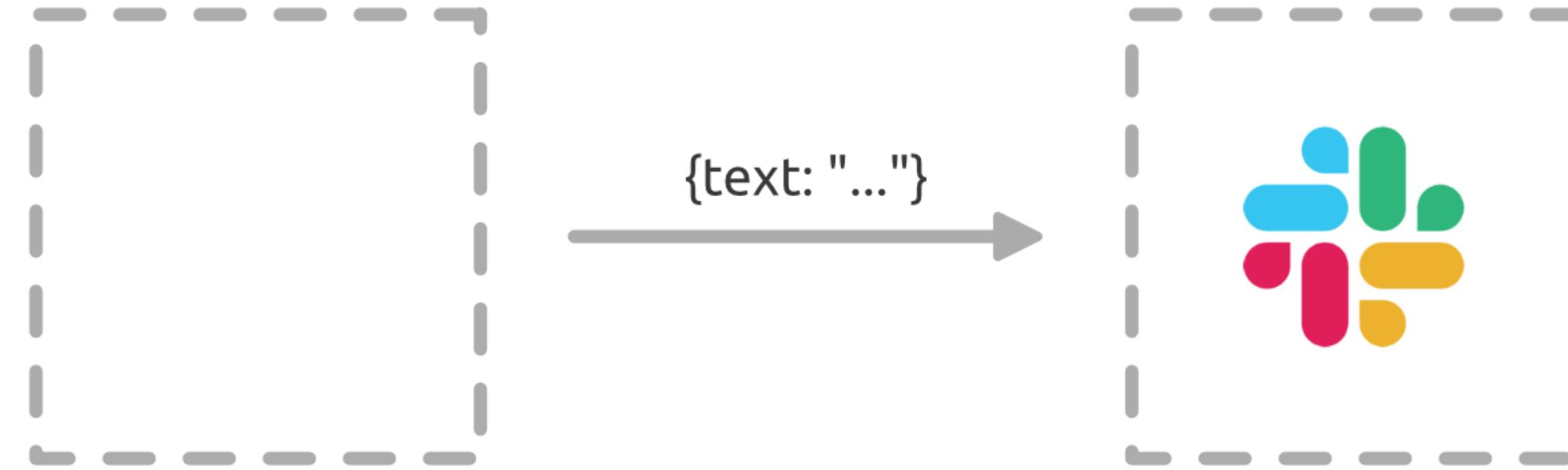
const slackService = require('./services/slack-incoming');
const actions = [
    require('./actions/salesData'),
    // ...
];

slackService.init(app);
actions.forEach((action) => {
    action(app, slackService);
} )

app.listen(port);
```

# Nachrichten in einen Channel posten

```
'https://hooks.slack.com/services/XXXX/XXXXXX'
```



# Slack-Nachrichten per URL einrichten

Webhook URL	Channel	Added By
<a href="https://hooks.slack.com/services/TEHML1">https://hooks.slack.com/services/TEHML1</a> <button>Copy</button>	#sales	Niklas Dec 3, 2018
<a href="https://hooks.slack.com/services/TEHML2">https://hooks.slack.com/services/TEHML2</a> <button>Copy</button>	#development	Niklas Dec 10, 2018
<a href="https://hooks.slack.com/services/TEHML3">https://hooks.slack.com/services/TEHML3</a> <button>Copy</button>	#random	Niklas Dec 30, 2018
<a href="#">Add New Webhook to Workspace</a>		

# Wie lief es gestern?

```
const process = require('process');
const axios = require('axios');

async function triggerSalesDataMessage() {
  axios.post(
    `https://hooks.slack.com/services/${process.env.SLACK_SALES_KEY}`,
    {
      text: `*Gestrigen Verkaufszahlen 🎉*` +
        '\n' +
        '\n' +
        'Hardwarebestellungen: *15.720€*\n' +
        'Softwareverkäufe: *15.720€*',
    }
  ).catch(console.log);
}
```

```
module.exports = function (app) {
  app.get('/sales-data', (req, res) => {
    triggerSalesDataMessage();

    res.redirect('/');
  });
}
```

# Anwendungsbeispiele

**Thomas D3V Bot** APP 1:55 PM**Neue Bestellung 41,41 €**

#23997 am 23.1.2019 12:54

Vorkasse - Offen

DHL Versand

niklas.buechner@viison.com

**Rechnung**

Pickware GmbH

Herr Niklas Büchner

Goebelstr. 21

64295 Darmstadt

**Positionen****1 x 29,90 €**

BarcodeScanner Schutzhülle iPod Touch 6G / iPhone 5 / 5s / SE

Shop Suche

**Mute Bot (LT10)** APP 11:45 AM**Heutiges Suppenangebot:**

- Bohneneintopf mit „Wienerle“
- Kartoffel-Feta-Möhren Suppe mit Chili und Koriander

**Braustübl Bot** APP 11:45 AM**Tagesangebot: Südfranzösische Bratwurst in Pfefferjus und Kroketten****Alternative: Kassler-Kamm in Bratenjus mit Fasssauerkraut und Butterkartoffeln****Vegetarisch: Würziger Flammkuchen mit Äpfeln, Zwiebeln und Handkäs****Versand**

Pickware GmbH

Herr Niklas Büchner

Goebelstr. 21

64295 Darmstadt

# Die neue Webseite deployen



**API Introduction** APP 11:48 PM

Outstanding Deployment

Would you like to deploy the new website now?

[Deploy new website](#)

[Deploy old website](#)

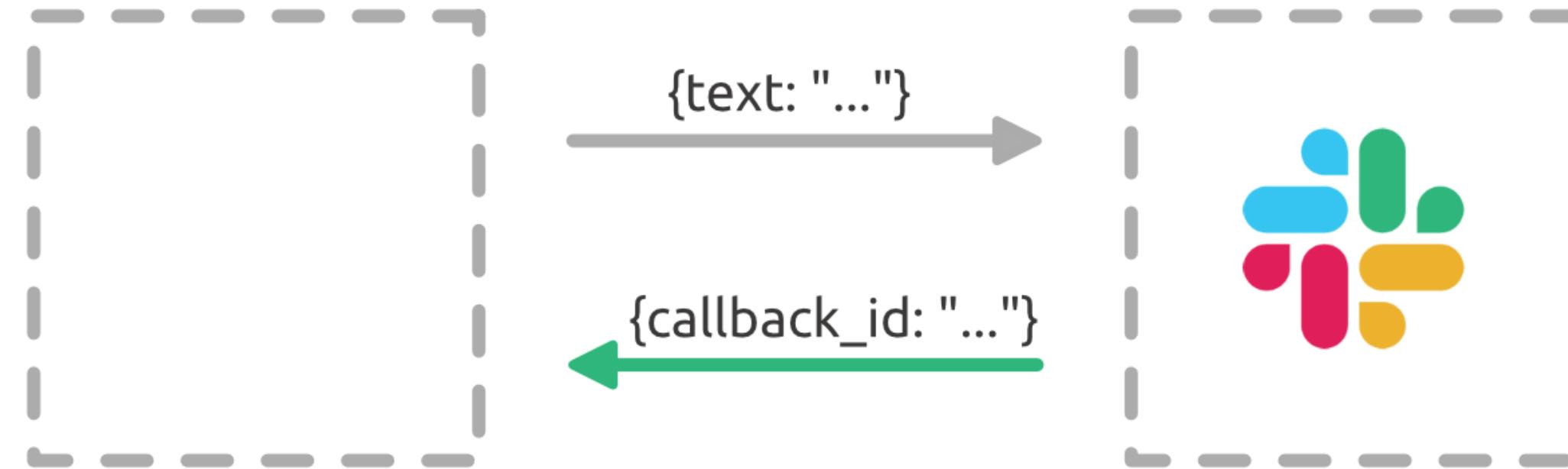
# Die neue Webseite deployen

```
'https://hooks.slack.com/services/' + process.env.SLACK_DEVELOPMENT_KEY
```

```
{
  text: '*Outstanding Deployment*',
  attachments: [ {
    text: 'Would you like to deploy the new website now?',
    fallback: 'Oops, looks like we can not deploy the site 😱',
    callback_id: 'deploy-website',
    actions: [ {
      name: 'deploy',
      text: 'Deploy new website',
      type: 'button',
      value: 'deploy',
    }, {
      name: 'deploy',
      text: 'Deploy old website',
      type: 'button',
      value: 'later',
    } ]
  } ],
}
```

# Interaktive Antwort

```
'https://hooks.slack.com/services/XXXX/XXXXXX'
```



```
'https://your.domain.example/slack-incoming'
```

# Interaktive URL definieren

The screenshot shows the Slack API Settings interface. At the top left is a navigation bar with a blue icon and the text "API Introduction". Below it is a dropdown menu. To the right of the dropdown is the title "Interactive Components".

The main area is divided into sections:

- Settings**: A sidebar with links to "Basic Information", "Collaborators", "Install App", and "Manage Distribution".
- Features**: A sidebar with links to "Incoming Webhooks", "Interactive Components" (which is highlighted with a blue background), and "Slash Commands".
- Interactivity**: A section with a green "On" toggle switch. Below it is the text: "Any interactions with actions, dialogs, message buttons, or message menus will be sent to a URL you specify. [Learn more.](#)".
- Request URL**: A text input field containing the URL "https://your.domain.example/slack-incoming". Below the input field is the explanatory text: "We'll send an HTTP POST request with information to this URL when users interact with a component (like a button or dialog)."

# Die neue Webseite deployen

```
var showOldWebsite = false;
slackService.webhook('deploy-website', 'deploy', (value) => {
    showOldWebsite = value === 'later';
}) ;
```

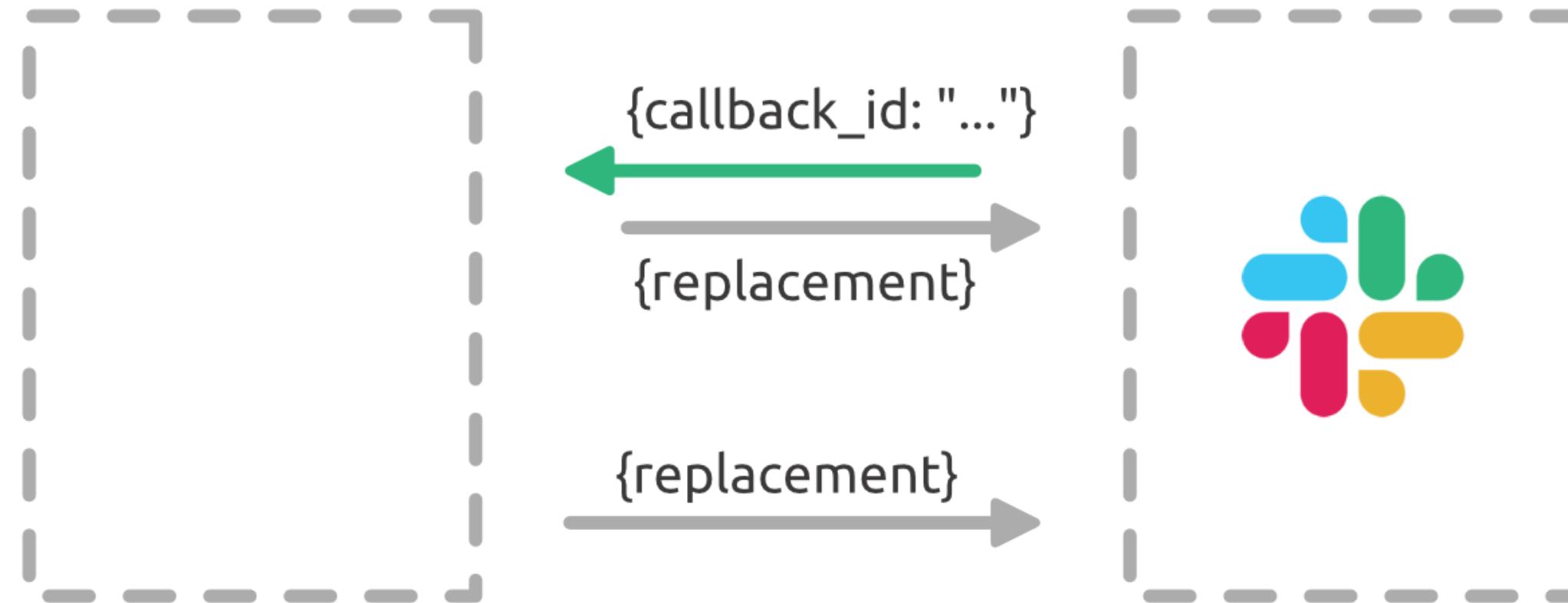
```
app.get('/website', (req, res) => {
    if (showOldWebsite) {
        res.render('old_site');
    } else {
        res.render('new_site');
    }
});
```

# Der Slack Service

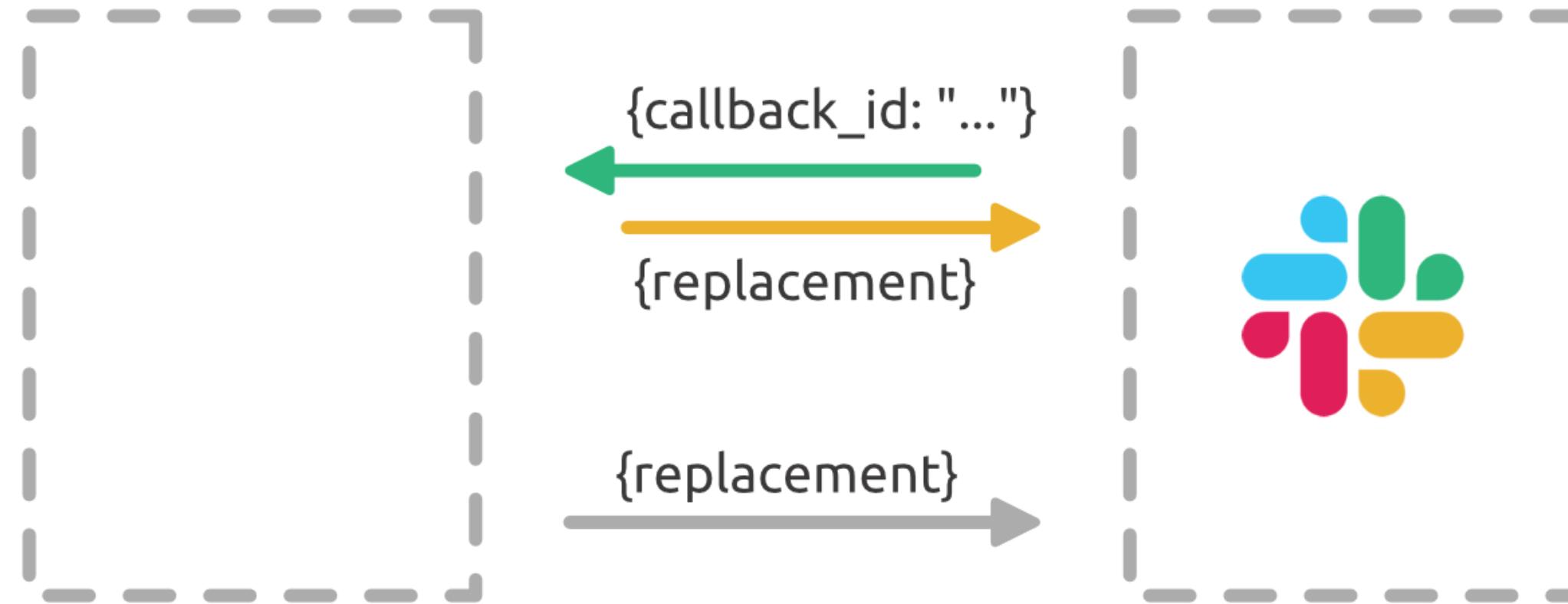
```
const incomingWebhooks = {};  
  
function webhook(callbackId, action, callback) {  
    incomingWebhooks[callbackId] =  
        incomingWebhooks[callbackId] || {};  
    incomingWebhooks[callbackId][action] = callback;  
}
```

```
app.post('/slack-incoming', (req, res) => {  
    const payload = JSON.parse(req.body.payload);  
  
    payload.actions.forEach(action => {  
        if (incomingWebhooks[payload.callback_id] &&  
            incomingWebhooks[payload.callback_id][action.name]) {  
            incomingWebhooks[payload.callback_id][action.name] (  
                action.value, payload, res  
            );  
        }  
    }) ;  
  
    res.status(200).end();  
});
```

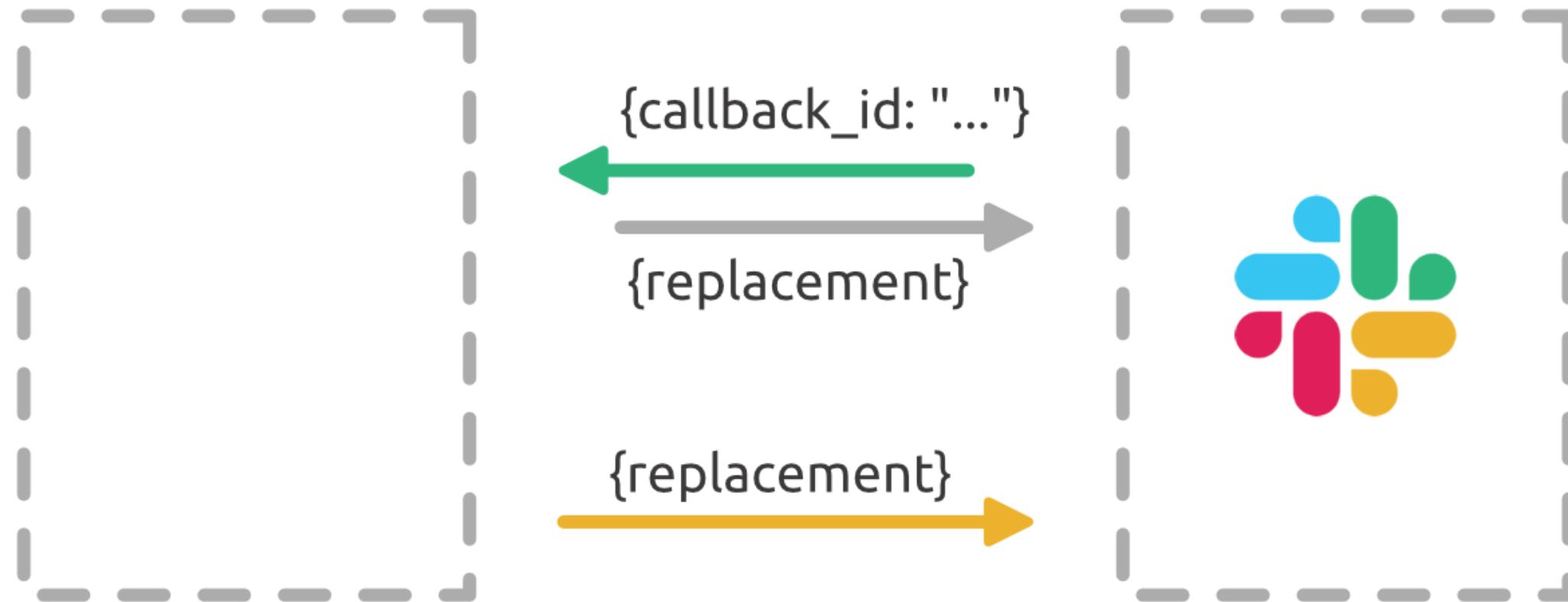
# Nachrichten ersetzen



# Nachrichten ersetzen



# Nachrichten ersetzen



# Nachrichten ersetzen

```
slackService.webhook('deploy-website', 'deploy', (value, payload) => {
  // ...
  if (!finalizeDeployment || showOldWebsite) {
    return;
  }

  axios.post(payload.response_url, {
    text: '*Outstanding Deployment*',
    attachments: [
      text: 'Would you like to deploy the new website' +
        ' now?\n<@' + payload.user.id +
        '> deployed the *new website*.'
    ],
  });
});
```

# Anwendungsbeispiele



**Pickware Shop Bot** APP 5:00 PM

Die Bestellung 23982 kann gepickt werden. Zeig unseren Kunden wie schnell mit Pickware versandt wird!

Wer kommissioniert?

Ich

# Neue GitHub Labels hinzufügen



**API Introduction** APP 12:16 AM

New GitHub Issue created

Please add labels to the issue #3

**Website looks really bad!!**

The new website I deployed via Slack looks terrible.

bug

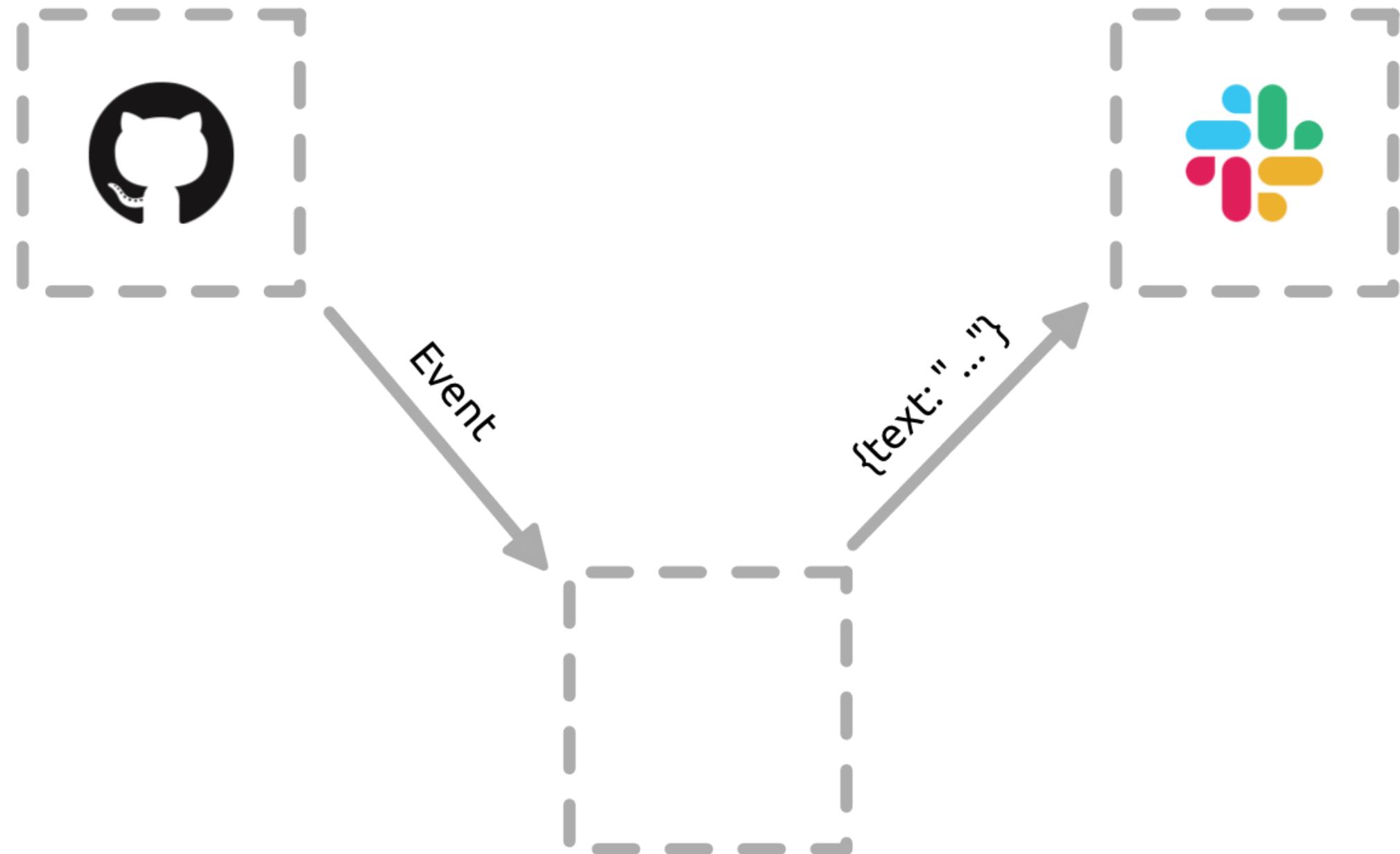
duplicate

enhancement

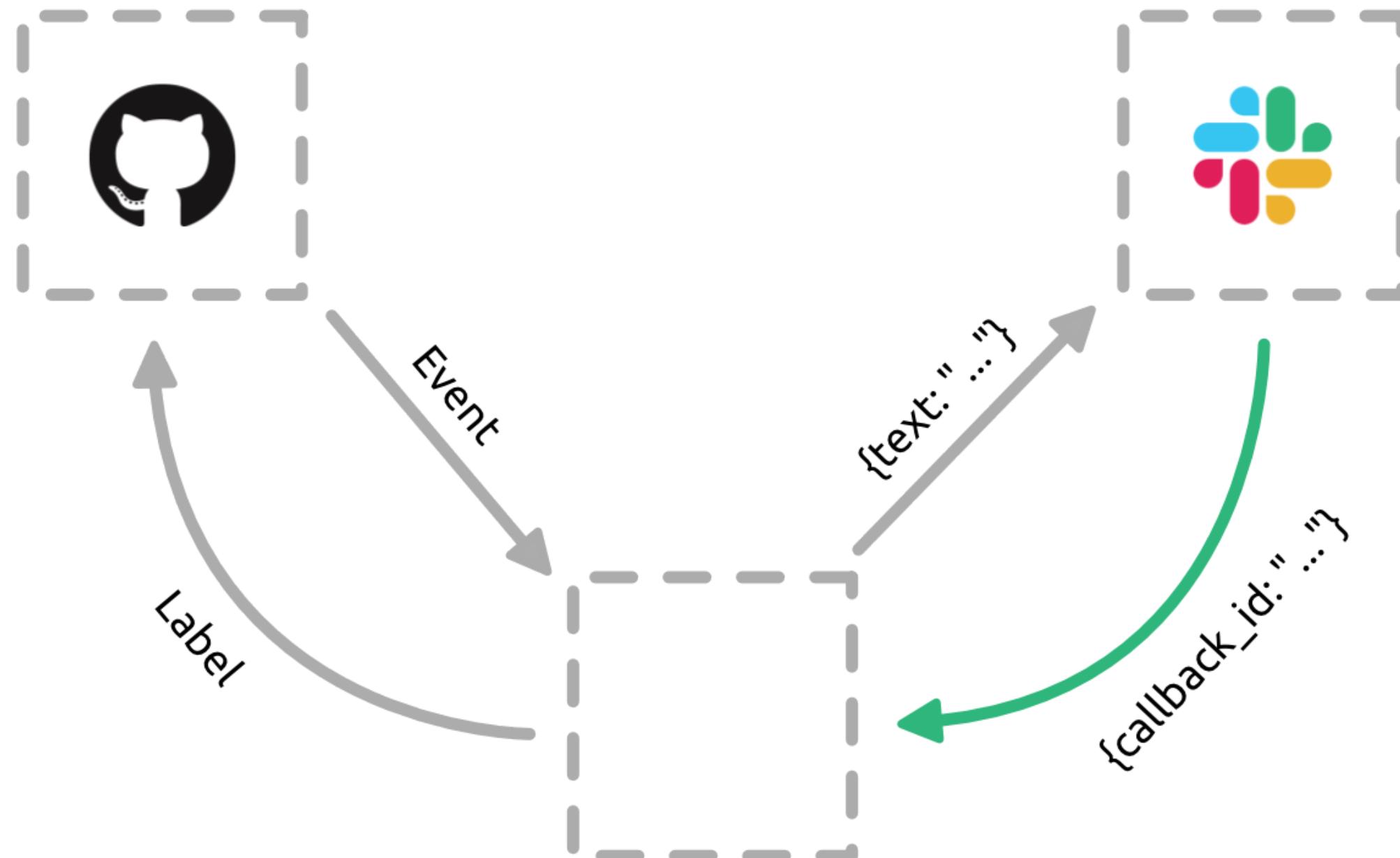
help wanted

good first issue

# GitHub Issue Event



# Label hinzufügen



# GitHub Events empfangen

The screenshot shows the GitHub Settings interface for managing webhooks. The top navigation bar includes links for Code, Issues (3), Pull requests (0), Projects (0), Wiki, Insights, and Settings (highlighted with an orange border). On the left, a sidebar menu lists Options, Collaborators, Branches, Webhooks (selected and highlighted with an orange border), Integrations & services, and Deploy keys. The main content area is titled "Webhooks" and contains a descriptive paragraph about what webhooks are and how they work. A single webhook entry is listed: "https://your.domain.example/label-event (issues)". To the right of this entry are "Edit" and "Delete" buttons.

Code Issues 3 Pull requests 0 Projects 0 Wiki Insights Settings

Options  
Collaborators  
Branches  
**Webhooks**  
Integrations & services  
Deploy keys

## Webhooks

Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our [Webhooks Guide](#).

- <https://your.domain.example/label-event> (issues)  
[Edit](#) [Delete](#)

# GitHub Events verarbeiten

```
app.post('/label-event', (req, res) => {
  if (!req.body.issue) {
    res.send('NO');
    res.status(502).end();
  }

  return;
}

const issue = req.body.issue;
triggerGithubIssueMessage(issue.title, issue.body, issue.number);

res.send('YES');
res.status(200).end();
} );
```

# Vorhandene Labels auslesen

```
const query =  
`query ($repositoryOwner: String!, $repositoryName: String!) {  
  repository(owner: $repositoryOwner, name: $repositoryName) {  
    labels(first: 20) {  
      nodes {  
        id,  
        name,  
      }  
    }  
  }`;
```

# Vorhandene Labels auslesen

```
const response = await axios.post(
  'https://api.github.com/graphql', {
    query,
    variables: {
      repositoryOwner: owner,
      repositoryName: name,
    }
  }, {
    headers: {
      'Content-Type': 'application/json',
      'Authorization': ` token ${process.env.GITHUB_TOKEN}`,
    },
  } );
return response.data.data.repository.labels.nodes;
```

# Slack-Nachricht erstellen

```
{  
    text: '*New GitHub Issue created*\n'  
        + 'Please add labels to the issue #' + number + '\n',  
    attachments: [ {  
        text: '*' + labelTitle + '*\n' + labelMessage,  
        fallback: 'Oops, looks like we can\'t add labels',  
        callback_id: 'add-labels',  
        actions: [ {  
            name: 'labels',  
            text: 'enhancement',  
            type: 'button',  
            value: label.name + '-' + number,  
        } ],  
    } ]  
}
```

# Label in GitHub hinzufügen

```
const githubRepo = process.env.GITHUB_REPOSITORY;
await github.addLabel(githubRepo, number, name);
```

```
const githubToken = process.env.GITHUB_TOKEN;
async function addLabel(repo, number, label) {
  return axios.post(
    `https://api.github.com/repos/${repo}/issues/${number}/labels`,
    {
      labels: [label],
    },
    {
      headers: {
        'Content-Type': 'application/json',
        'Authorization': `token ${githubToken}`,
      },
    },
  );
}
```

# State in Nachrichten speichern

```
{  
    text: '*New GitHub Issue created*\n'  
        + 'Please add labels to the issue #' + number + '\n',  
    attachments: [ {  
        text: '*' + labelTitle + '*\n' + labelMessage,  
        fallback: 'Oops, looks like we can\'t add labels',  
        callback_id: 'add-labels',  
        actions: [ {  
            name: 'labels',  
            text: 'enhancement',  
            type: 'button',  
            value: label.name + '-' + number,  
        } ],  
    } ]  
}
```

# Anwendungsbeispiele



**Pollo** APP 2:02 PM

@Niklas Büchner has a poll for you!

How do you like my presentation?



Delete Poll

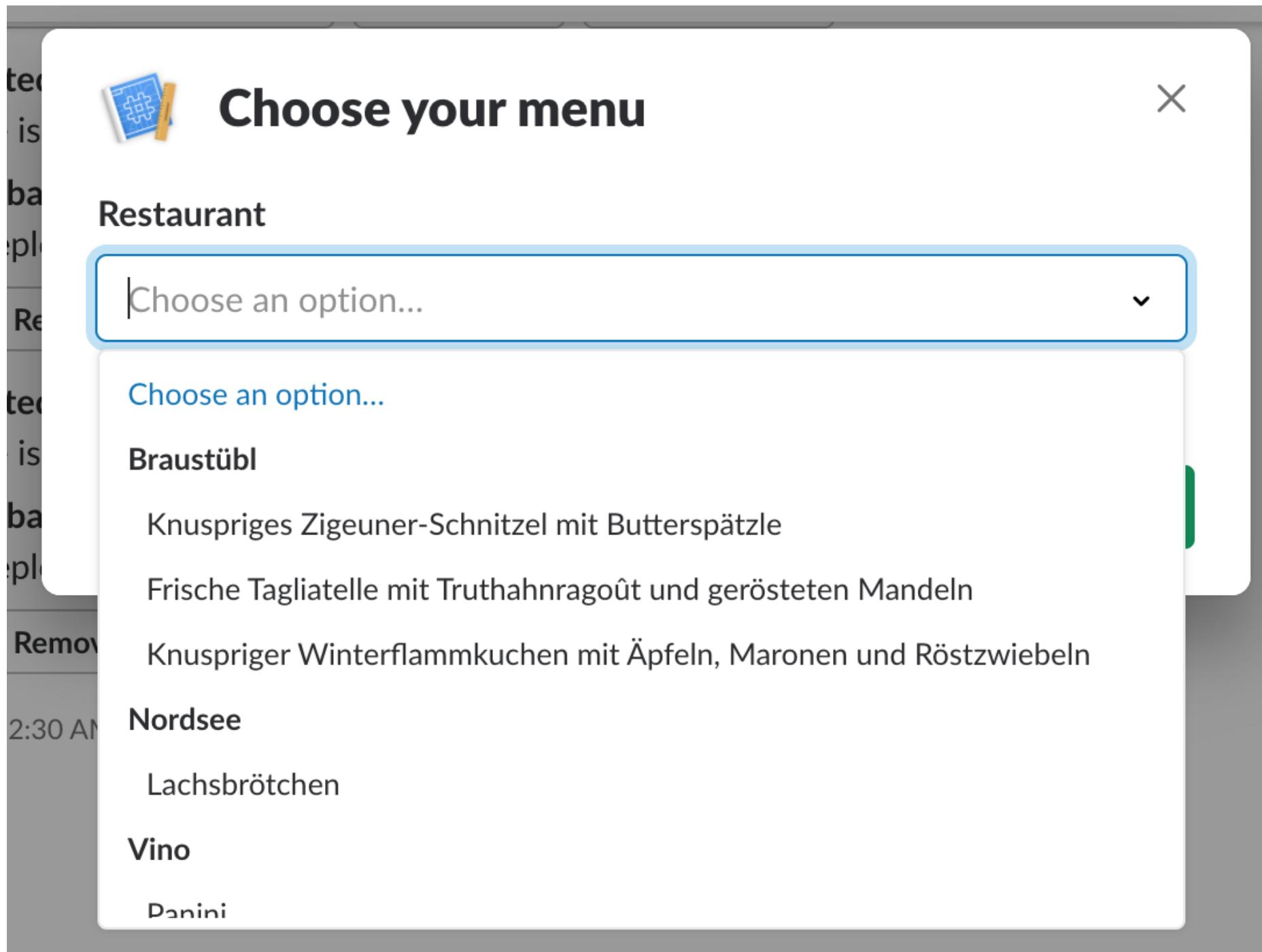


**Türklingel (Nello)** APP 1:34 PM

@here Es hat geklingelt! Kann bitte jemand die Tür öffnen?

Tür öffnen

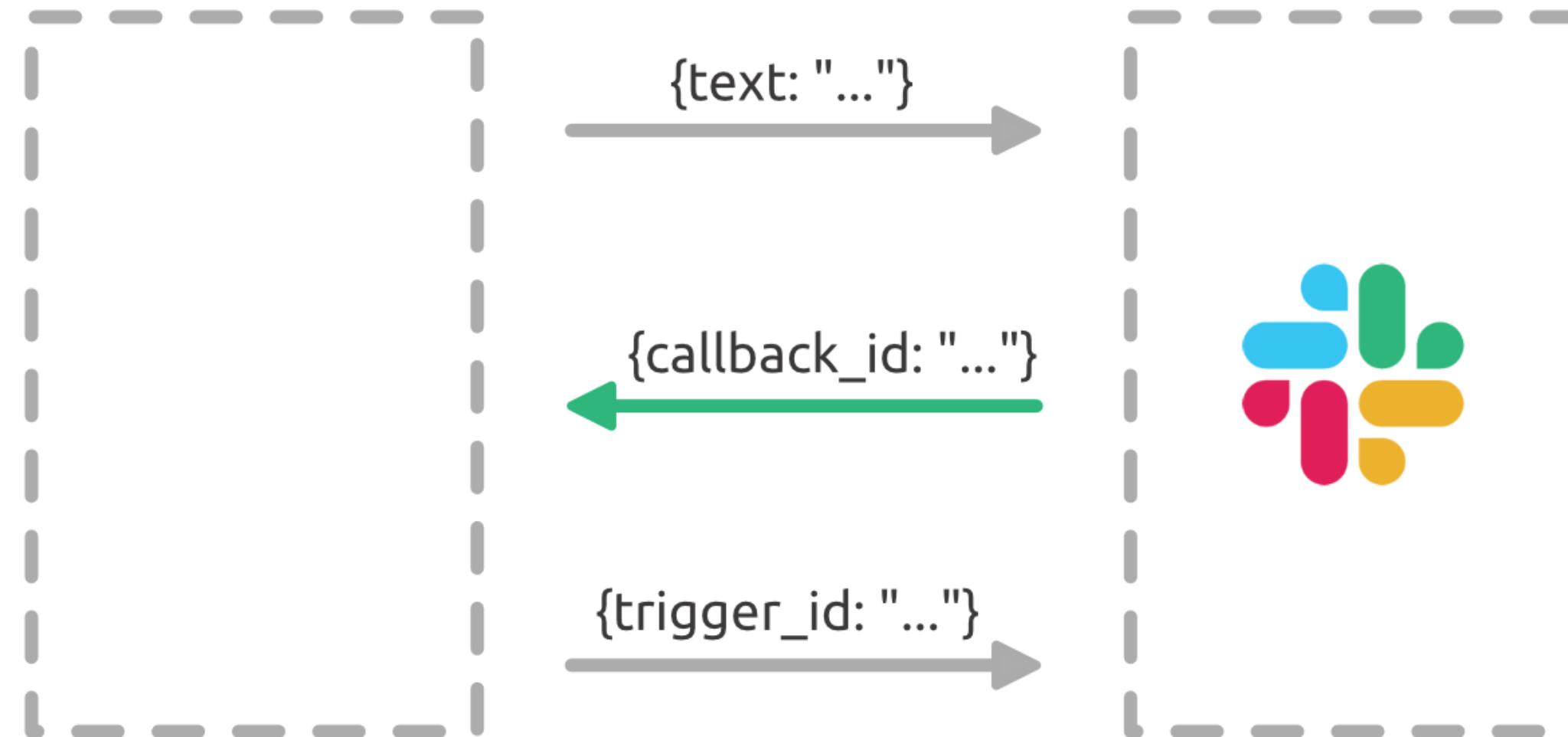
# Mittagessen per Dialog



# Mittagessensnachricht

```
{  
  text: '*Order lunch*',  
  attachments: [ {  
    text: '',  
    callback_id: 'order-lunch',  
    actions: [ {  
      name: 'order',  
      text: 'Order lunch',  
      type: 'button',  
      value: 'order'  
    } ]  
  } ],  
}
```

# Dialog öffnen



# Dialog öffnen

```
axios.post('https://slack.com/api/dialog.open', {
  trigger_id: payload.trigger_id,
  dialog: {
    callback_id: "lunch-order",
    title: "Choose your menu",
    submit_label: "Order",
    notify_on_cancel: true,
    elements: [...]
  }
}, {
  headers: {
    Authorization: "Bearer " + slackOAuthToken,
  },
}) .catch(console.log);
```

# Select zusammenstellen

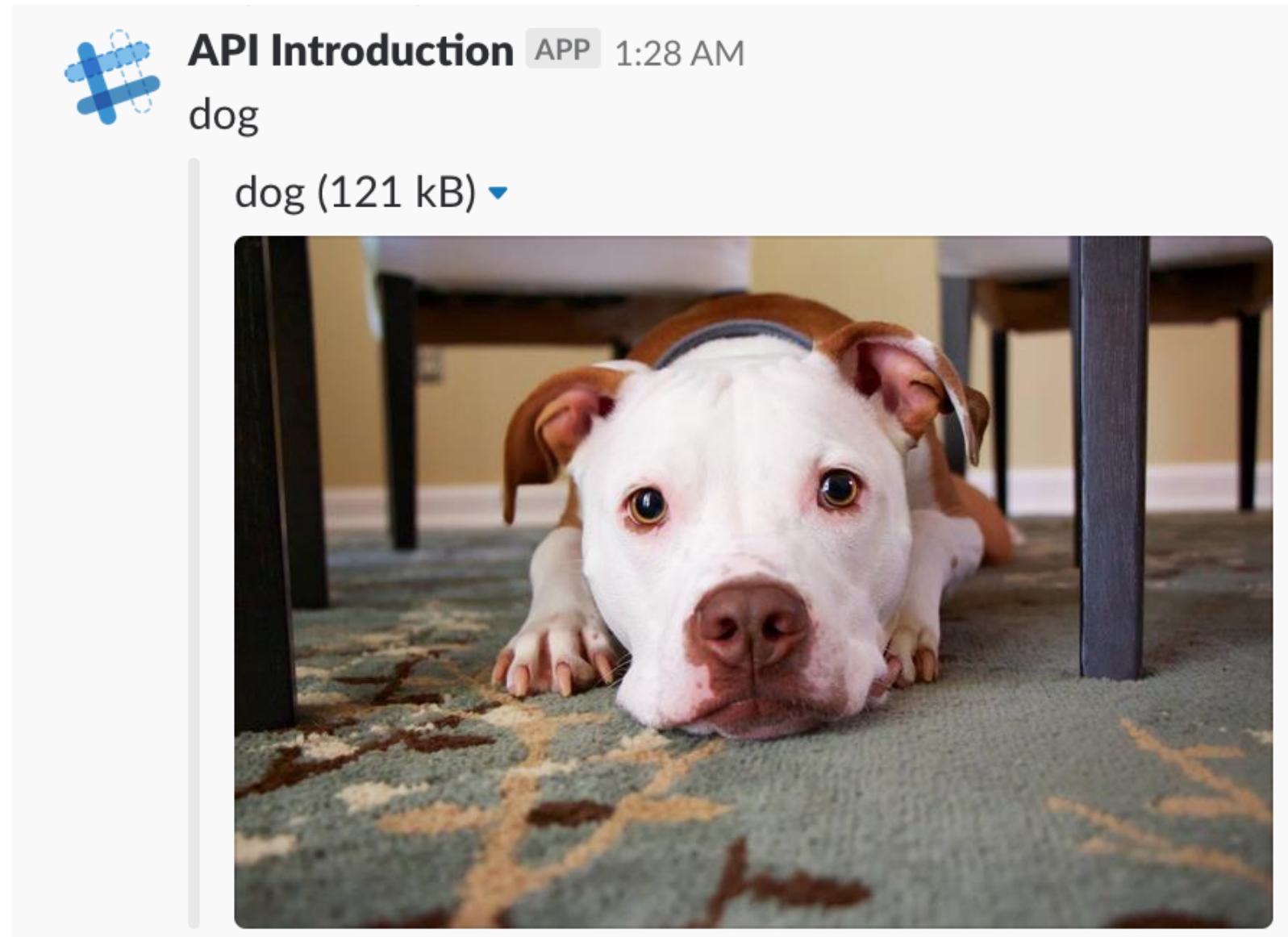
```
elements: [ {  
    type: "select",  
    label: "Restaurant",  
    name: "restaurant",  
    option_groups: [ {  
        label: "Braustübl",  
        options: [ {  
            label: "Knuspriges Zigeuner-Schnitzel mit Butterspätzle",  
            value: "regular",  
            // ...  
        } ],  
        // ...  
    } ],  
    // ...  
},  
]
```

# Antwort verarbeiten

```
slackService.webhook('lunch-order', null, sendLunchConfirmation);

async function sendLunchConfirmation(value, payload) {
  axios.post('https://hooks.slack.com/services/' + process.env.SLACK_DEVELOPM
    text: '*Lunch ordered*\n'
      + 'Ordered ' + payload.submission.restaurant + '\nYour order will a
  } ).catch(console.log);
}
```

# Unsplash Command



+

/unsplash dog

# Command registrieren

The screenshot shows the configuration interface for a Slack app named "Sales Alert". The left sidebar has sections for "Settings" (Basic Information, Collaborators, Install App, Manage Distribution) and "Features" (Incoming Webhooks, Interactive Components, **Slash Commands**, OAuth & Permissions). The main area is titled "Slash Commands" and contains a table with one entry: "/unsplash" (Description: Random image from unsplash), with edit and delete icons. A "Create New Command" button is at the bottom.

Sales Alert ▾

**Slash Commands**

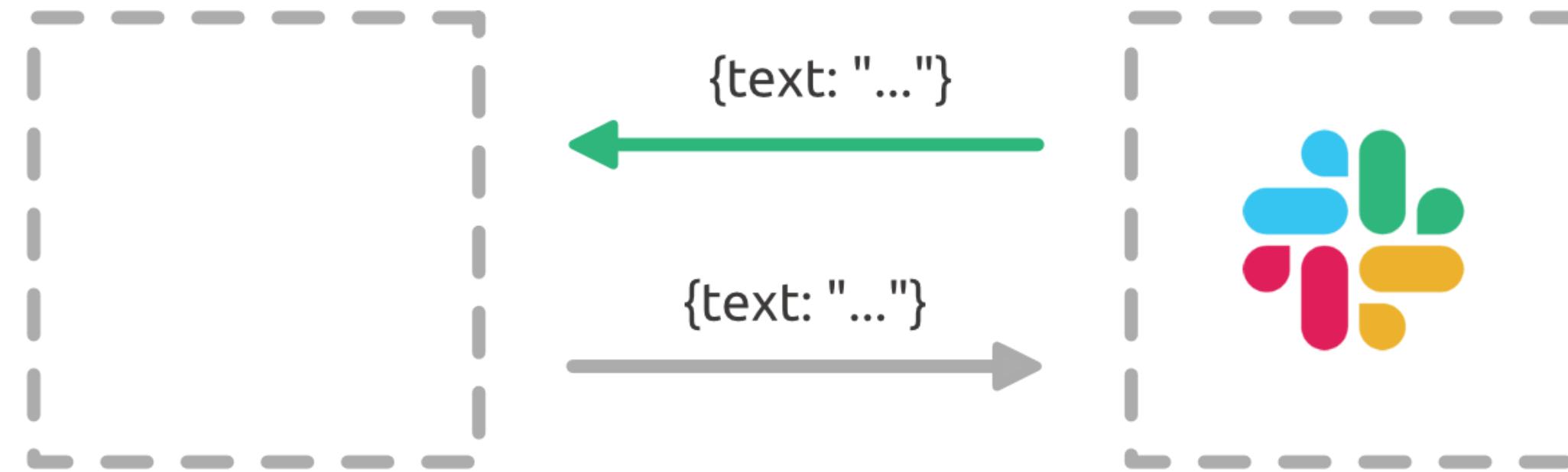
Commands enable users to interact with your app from within Slack. [Learn more.](#)

Name	Description	
/unsplash	Random image from unsplash	

**Create New Command**

# Slash Command verarbeiten

```
'https://your.domain.example.unsplash'
```



# Command initialisieren

```
app.post('/unsplash', (req, res) => {
  const body = req.body;
  getRandomImageFromUnsplash(body.response_url, body.text);

  res.status(200).end();
}) ;
```

# Bild laden

```
async function getRandomImageFromUnsplash(responseUrl, query) {  
  const result = await axios.get(  
    'https://api.unsplash.com/photos/random?query=' + query,  
    {  
      headers: {  
        'Authorization': 'Client-ID '  
          + process.env.UNSPLASH_ACCESS_KEY  
      },  
    }  
  ) .catch(console.log);  
  
  // ...  
}
```

# Bild anzeigen

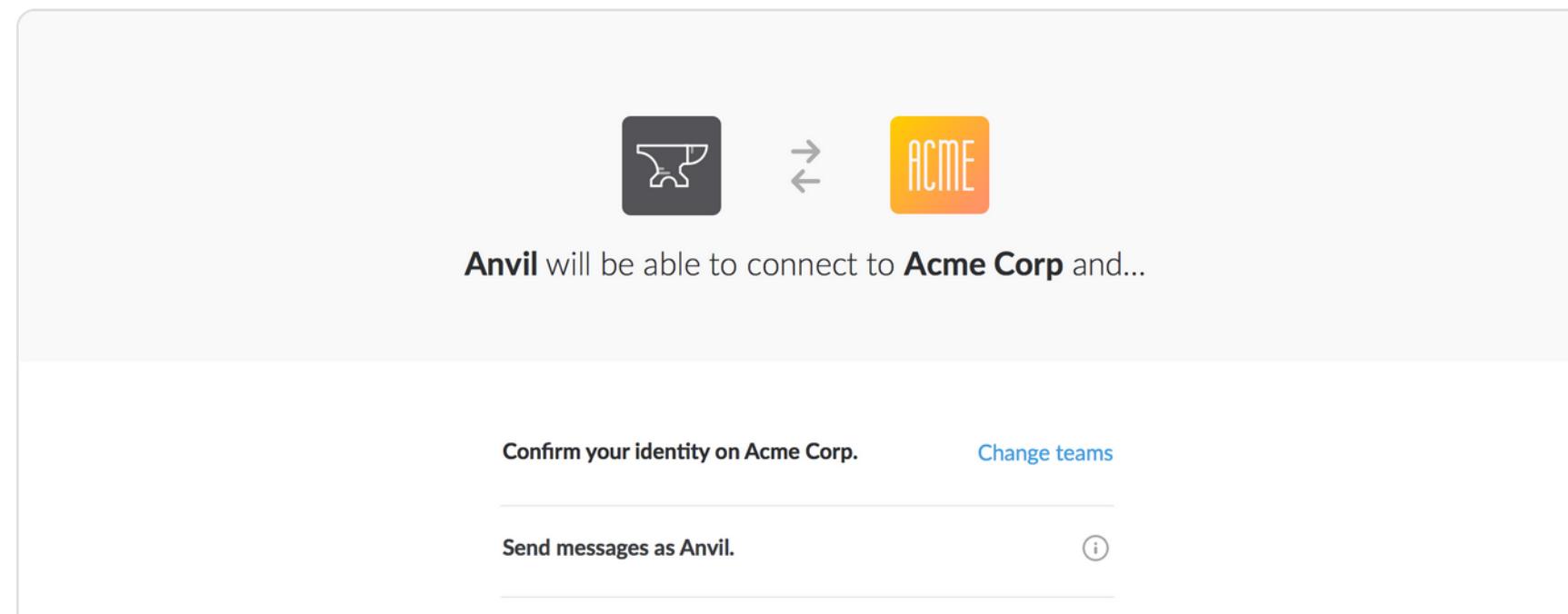
```
async function getRandomImageFromUnsplash(responseUrl, query) {  
    // ...  
  
    const randomImage = result.data.urls.regular;  
    axios.post(responseUrl, {  
        text: query,  
        attachments: [ {  
            text: query,  
            image_url: randomImage,  
        } ],  
    }) .catch(console.log);  
}
```

# App als SaaS bereistellen

## OAuth Permission scopes

OAuth scopes let you specify exactly how your app needs to access a Slack user's account. As an app developer, you specify your desired scopes in the initial [OAuth authorization request](#). When a user is responding to your OAuth request, the requested scopes will be displayed to them when they are asked to approve your request.

Slack's system of OAuth permission scopes governs usage of [Slack Apps](#) and their use of the [Web API](#), [Events API](#), [RTM API](#), [Slash Commands](#), and [Incoming Webhooks](#).



# Vielen Dank für eure Aufmerksamkeit



[/niklasbuechner/slack-api-](#)  
[presentation](#)

[niklas.buechner@pickware.com](mailto:niklas.buechner@pickware.com)

