CI/CD Flow

An end-to-end CI/CD flow automates the software development lifecycle, from code changes to deployments, ensuring faster and more reliable releases. It involves several stages, including source, build, test, and deploy.

Here's a detailed look at the end-to-end CI/CD flow:

1. Source:

Code changes are made and merged into a shared code repository.

This stage can be triggered automatically by code changes, manually by a user, or on a schedule.

2. Build:

The CI/CD pipeline compiles the source code, resolves dependencies, and packages the application.

This creates a deployable artefact, often a Docker image.

3. Test:

Automated tests are run to validate the code and ensure it meets quality standards.

This stage can include unit tests, integration tests, and acceptance tests.

4. Deploy:

The built and tested artifact is deployed to different environments, like staging or production.

Continuous Delivery (CD) can automate the deployment to chosen environments, while Continuous Deployment (CD) automatically releases updates to production.

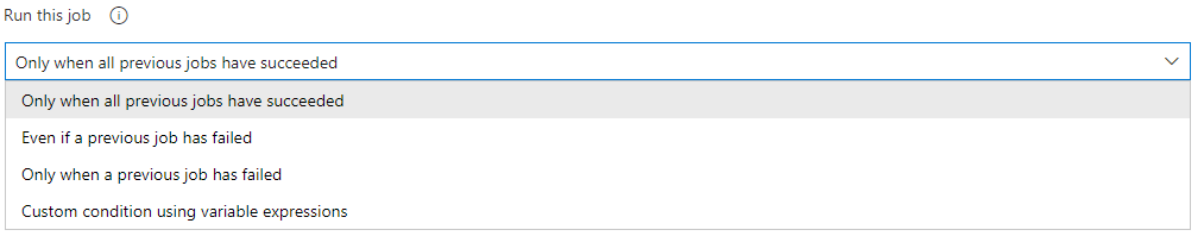
5. Continuous Feedback:

The CI/CD pipeline provides continuous feedback to developers, allowing them to identify and fix issues quickly.

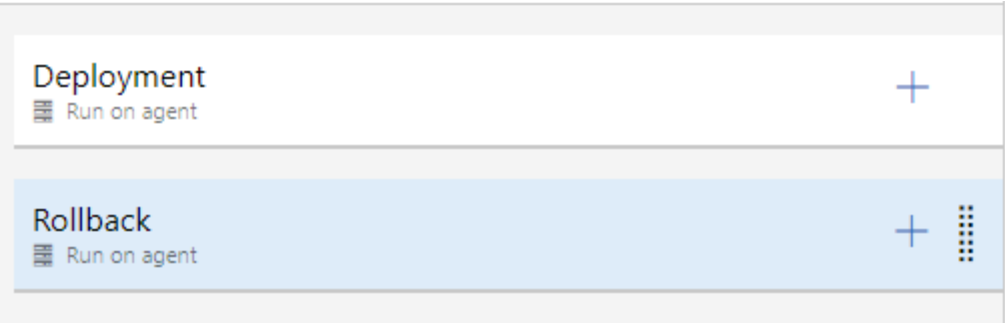
This feedback loop enables faster and more reliable releases.

Add rollback strategy in pipeline

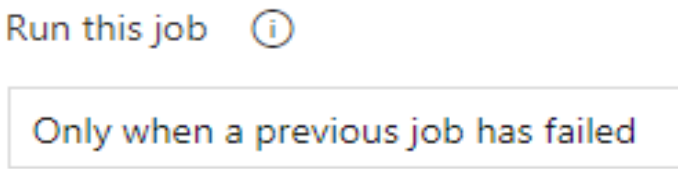
How to implement rollback strategies in Azure Pipelines:



This means you should have, at a minimum, one job for your deployment and one for your rollback if something goes wrong:



The Rollback job will have this execution condition:



To set up deployment notifications via Slack or email, you'll typically use webhook integrations and/or platform-specific notification settings. You can configure Slack notifications to receive deployment status updates in a designated channel, or configure email notifications to be sent to specific individuals or teams.

Elaboration:

1. Webhook Integrations:

Slack:

You can use webhooks from Netlify, Humanitec, or other platforms to send deployment-related information to a Slack channel.

HTTP Post Request:

This method allows you to send data to any URL, including a Slack webhook, for receiving notifications.

Process:

You'll usually create a webhook within your deployment platform (e.g., Netlify, Render, DeployHQ) and configure it to send data to your Slack channel.

2. Platform-Specific Notification Settings:

Email:

Many deployment platforms offer email notification options, allowing you to receive deployment status updates via email.

Render:

For example, Render allows you to choose between email, Slack, or both for notifications, and also offers customization options for specific services.

Netlify:

Netlify Docs offers both Slack and email notification options, with email available on Pro and Enterprise plans.

3. Configuring Slack Notifications:

Slack App:

You can add a Slack app like Incoming Webhooks to your Slack workspace and configure it to receive notifications from your deployment platform.

Workflow Builder:

Slack's Workflow Builder can be used to automate sending messages to a channel when a deployment is complete or fails.

Customization:

You can customize the messages sent to Slack, include relevant information about the deployment (e.g., deployment URL, environment), and configure the frequency of notifications.

4. Email Notifications:

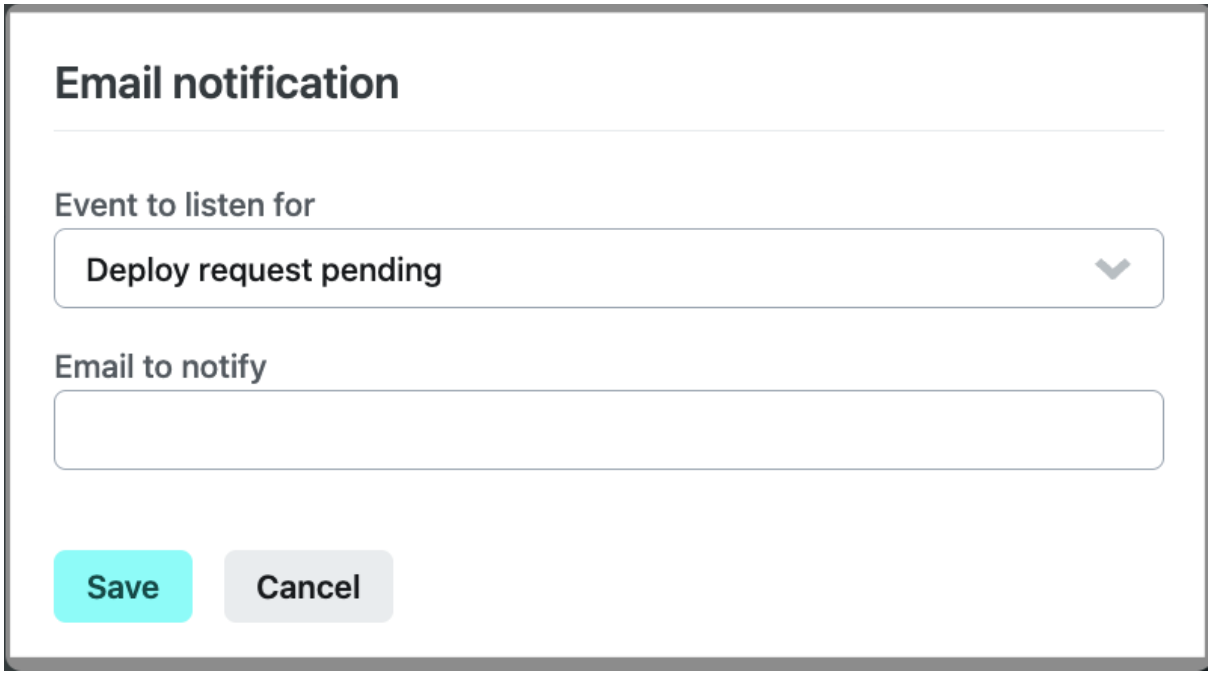
Recipient:

Specify the email address(es) that should receive deployment notifications.

Content:

The email will typically contain information about the deployment, such as its status, URL, and any errors encountered.

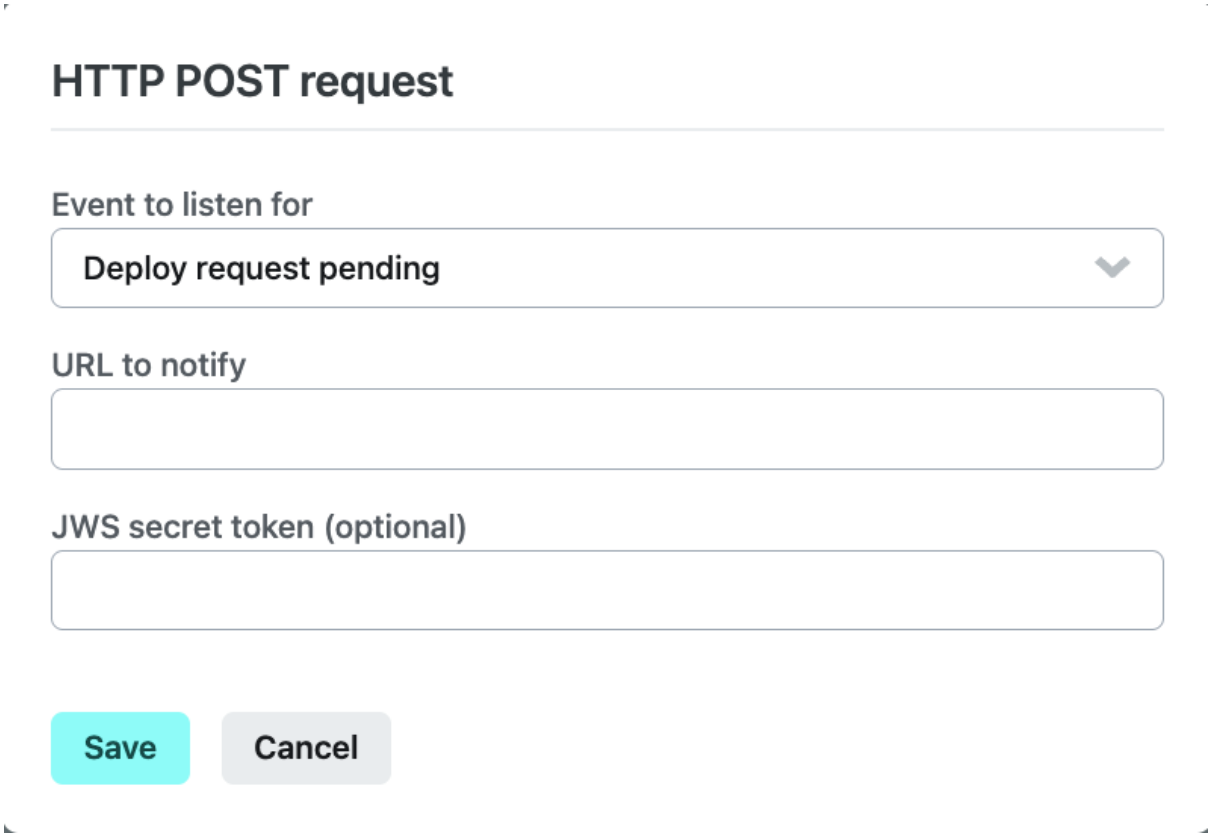
Email notifications



HTTP Post Request

This type of notification works as an outgoing webhook and allows you to send event information to an arbitrary URL using a POST request.

The body of the outgoing webhook request will have a JSON representation of the object relevant to the event.



Payload signature:

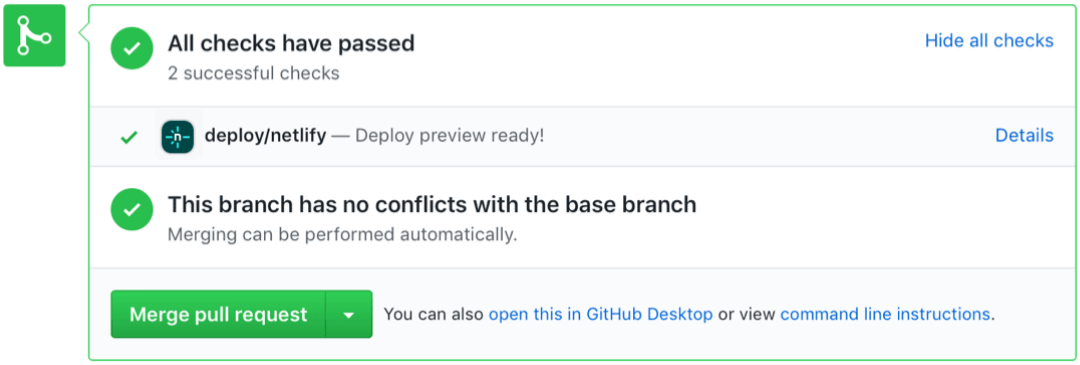
If you provide a JWS secret token for an outgoing webhook, Netlify will generate a JSON Web Signature(JWS) and send it along with the notification in the request header X-Webhook-Signature.

We include the following fields in the signature's data section:

iss: always sent with value netlify, identifying the source of the request

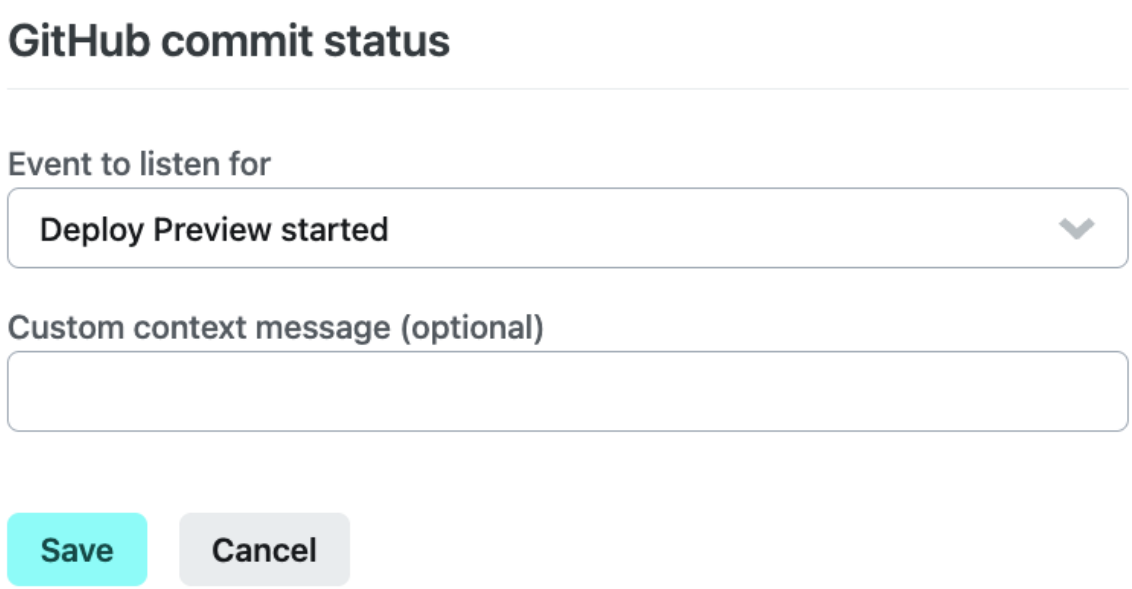
sha256: the hexadecimal representation of the generated payload's SHA256

GitHub commit statuses:



add, remove, or edit them in Project configuration > Notifications > Deploy notifications.

The settings include a field for a custom message, which will replace the “Deploy preview ready!” message that displays by default.



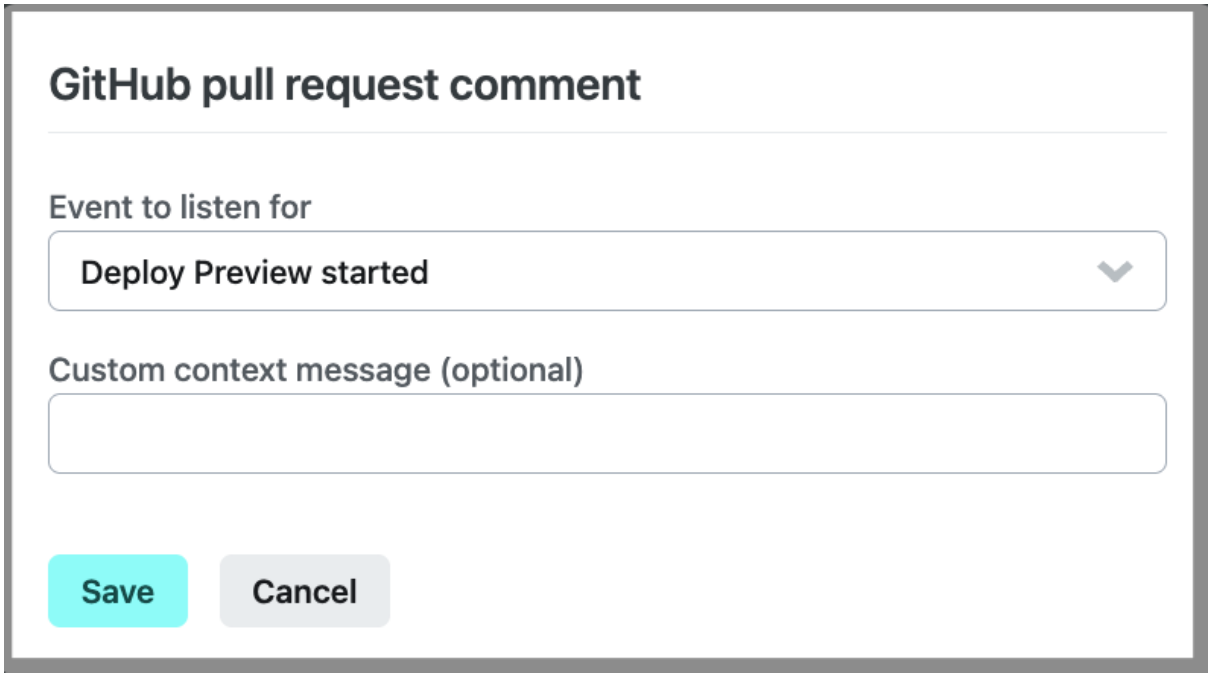
GitHub pull request comments:

This type of notification adds a comment to your GitHub pull requests indicating the status of the associated deploy and providing a link to the Deploy Preview when ready. If you append more commits to a

pull request, this notification will update the comment to indicate status changes.

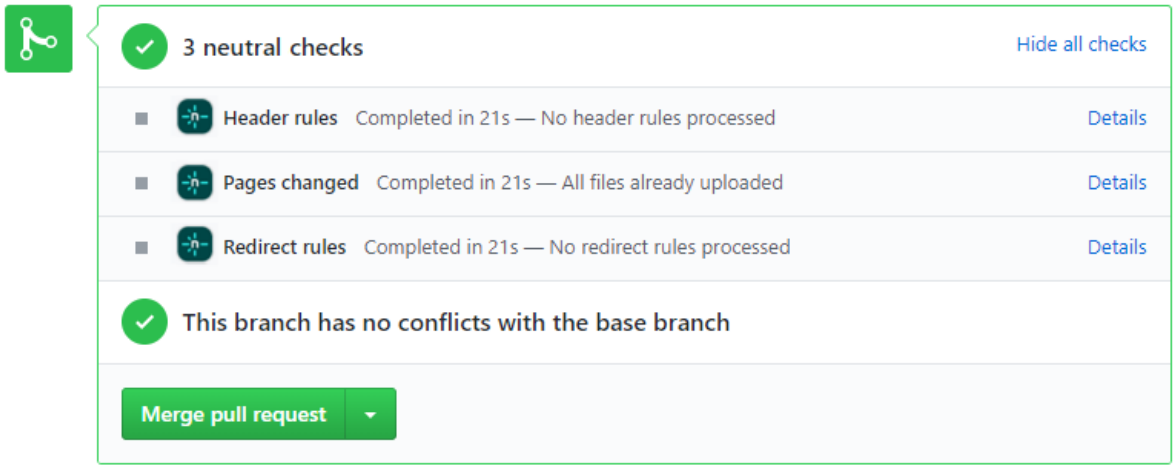
GitHub pull request comment notifications are added to all new GitHub-connected Netlify sites by default. You can add, remove, or edit them in Project Configuration> Notifications > Deploy notifications.

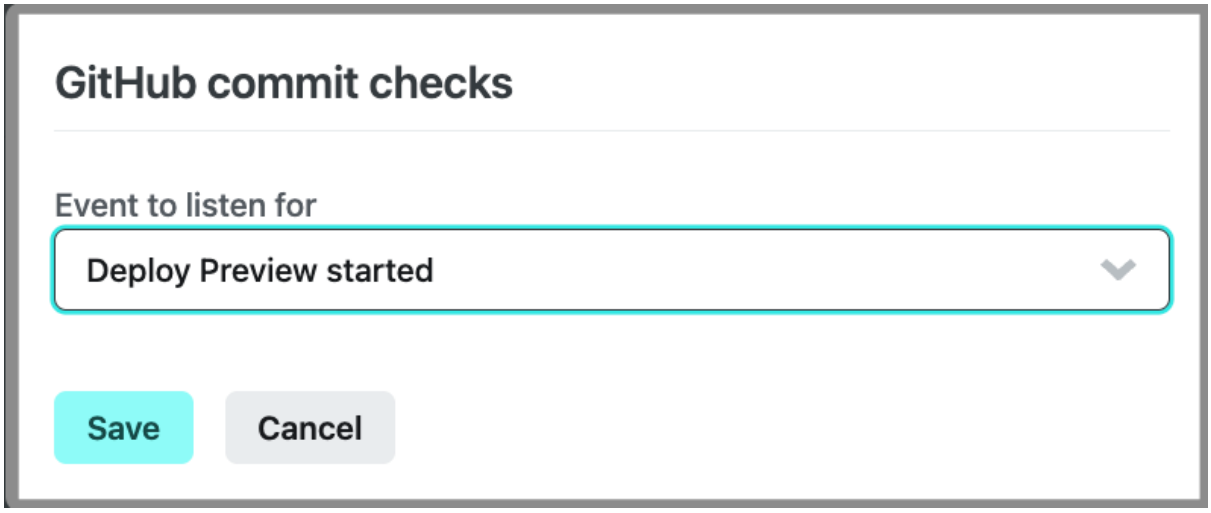
The settings include a field for a custom message, which will replace the “Deploy preview ready!” message that displays by default.



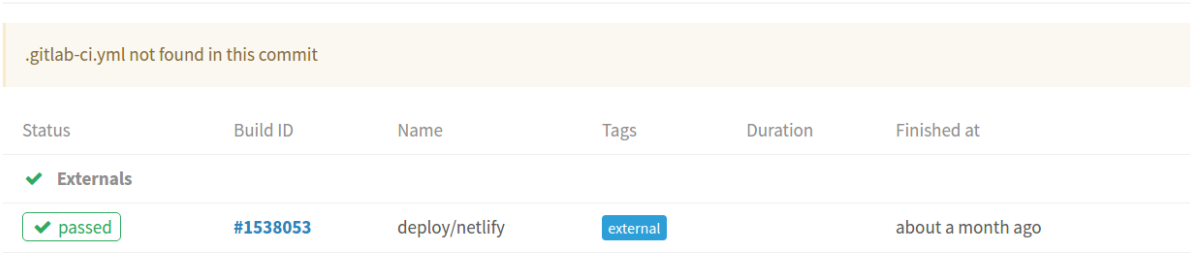
GitHub commit checks

This type of notification adds rich deploy information from you deploy summary to your GitHub pull requests and commit lists. This includes more detailed information in the Checks tab of your pull requests on GitHub.





GitLab commit statuses



GitLab commit status

Event to listen for

Deploy Preview started

Personal access token

How to generate a GitLab API access token ↗

Custom context message (optional)

API url for on-premise hosts (optional)

Save Cancel

Some Person @exampleperson · 6 minutes ago

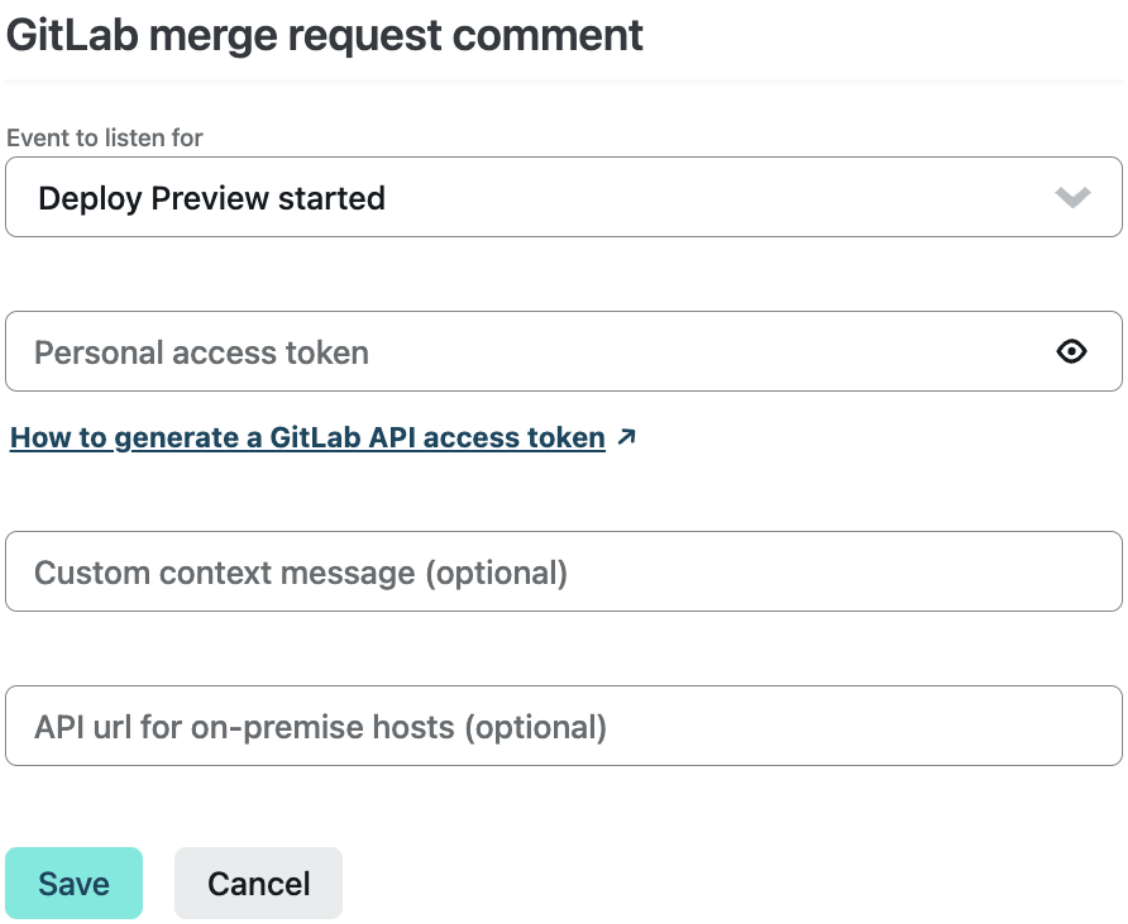
Author Owner

✅ Deploy Preview for *eloquent-bell-f8d16b* ready!

Name	Link
🔨 Latest commit	0f172bea
🔍 Latest deploy log	https://app.netlify.com/sites/eloquent-bell-f8d16b/deploys/6239ed256c93510009deb61c
😎 Deploy Preview	https://deploy-preview-2--eloquent-bell-f8d16b.netlify.app/
📱 Preview on mobile	▶ Toggle QR Code...

To edit notification comments on pull requests, go to your Netlify site settings.

Edited by Some Person 6 minutes ago



Troubleshoot GitLab deploy notifications

GitLab deploy notifications stop working

GitLab merge request comment

Event to listen for

Deploy Preview started

Personal access token

How to generate a GitLab API access token ↗

Custom context message (optional)

API url for on-premise hosts (optional)

Save Cancel