Ensuring proper reviews

with Pull Requests & Branch Protection rules

The power of **Pull Requests**

- Opened once a developer has **completed code changes**
- Ideal opportunity for **collaborative code review**
- Ability to **accept, reject and amend changes before merge**

Enforcing PRs with **Branch Protection rules**

Branch protection rules on the main branch in order to:

- Force new code to be merged with a **Pull Request**
- Ensure at least **1 Approval on the PR** (and dismiss them upon a PR update)
- Require **resolution of any conversation on the PR**

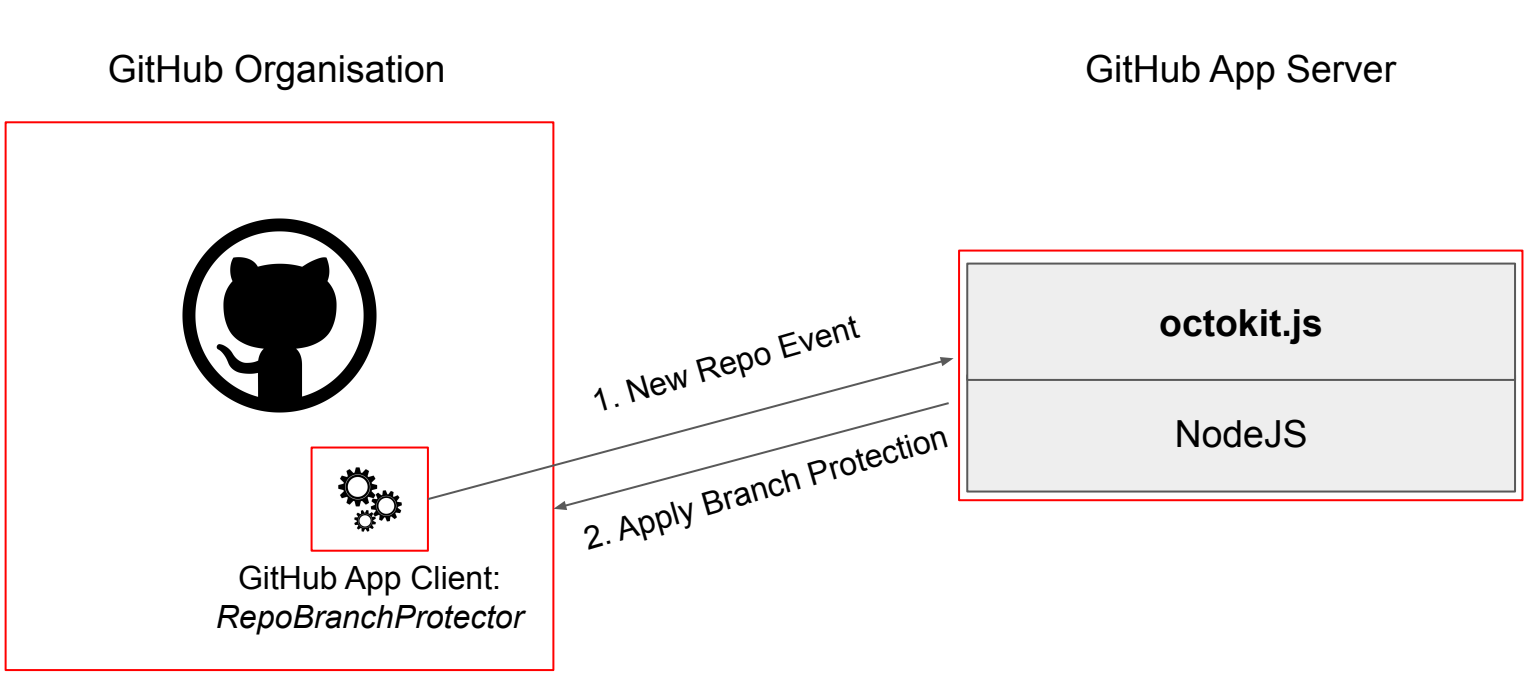
Let’s Automate it for Scale!

A GitHub App to automate protection

- Has its own identity
- Only specific permissions granted
- Leverages the GitHub API
- Subscribes and reacts to events via webhooks

A word on GitHub API and Libraries

- You can directly interact with the GitHub REST API (currently v3)
- Or use an official **Octokit library** instead
  - Simplified access and API call handling (failures, throttling, etc)
  - Available **officially** for **.NET, JS and Ruby** environments
  - Unofficially for a myriad others (open source)


GitHub Organisation
GitHub App Server
1. New Repo Event
2. Apply Branch Protection
octokit.js
NodeJS
GitHub App Client:
RepoBranchProtector

Advanced Security Review

Beyond manual PR reviews

- **Security scanning**: CodeQL for automated SAST scanning
- **Dependency reviews**: PR checks and Dependency alerts
- **Secret detection**

→ Let’s continue the conversation

Above features are enabled on public repos, licensed for private Enterprise repos. Read more:

https://github.com/features/security