happiest minds
The Mindful IT Company
Born Digital . Born Agile

GitHub

DevOps Management with GitHub

Table of Contents

Introduction to GitHub

Since its inception in 2008, GitHub has continuously transformed the way people code, making it easier to collaborate and develop elegant, disparate solutions for the market. GitHub has grown and evolved from a version control system to a social utility for programmers and finally to the place were code lives online.

It is a blessing in disguise for software developers by creating the most comprehensive open-source platform facilitating cloud storage for source code, codeshare, networking, publishing services, and code talks. Today millions of people around the globe use these repositories. GitHub has achieved an incredible growth, and collaboration with Microsoft has scaled it to a new height which has helped many users efficiently maintain complex codebases.

This whitepaper provides you with a detailed guide about GitHub, and how you can effectively deploy and manage your code using Git & GitHub.

What is GitHub?

GitHub is a well-known web-based open-source platform where users host git repositories. It is a highly used software version control and is helpful when more than one person is working on a project.

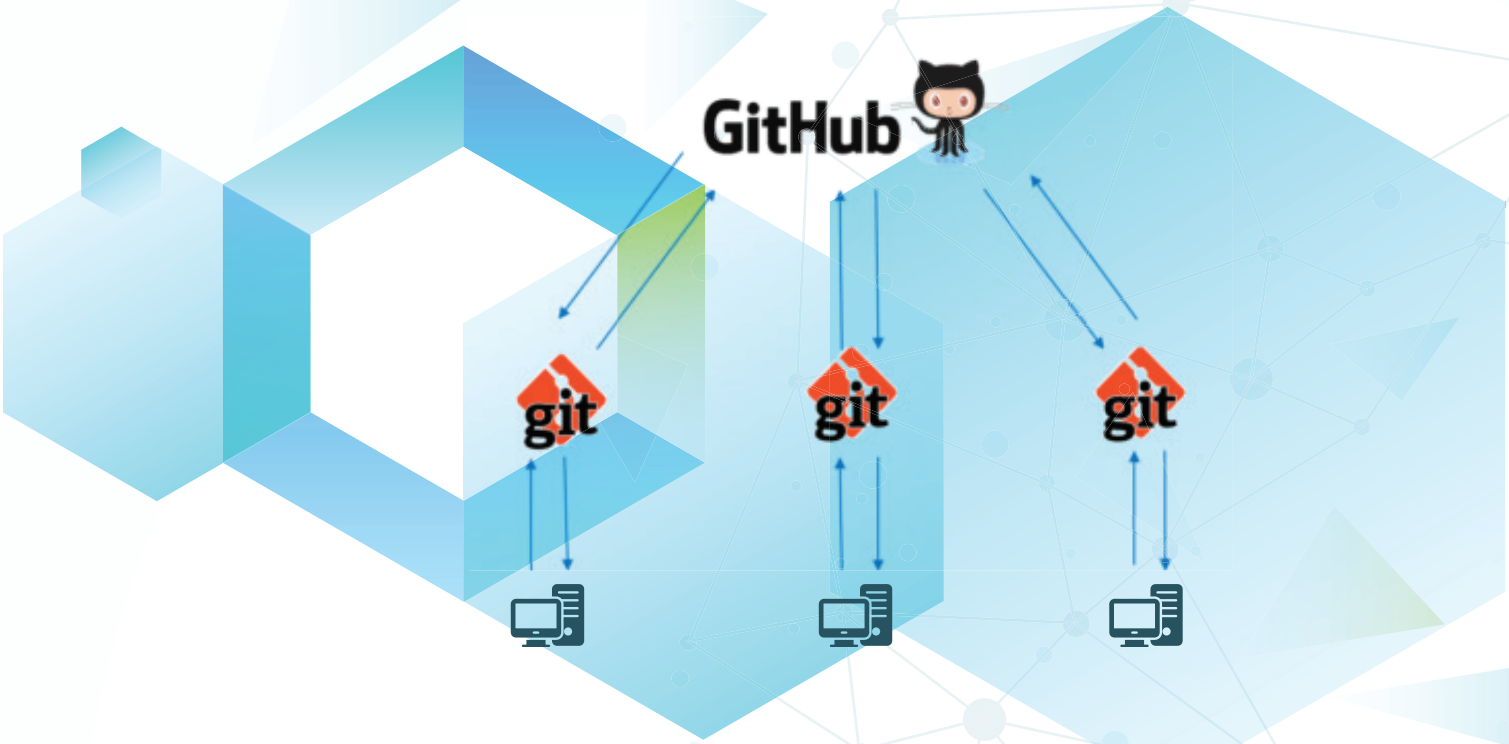
Why GitHub?

GitHub provides a centralized repository for the software development team to upload, edit, and manage code files to help them build a website. Everyone can update their codes simultaneously when working on the project. GitHub allows us to document the changes in codes files and reflect them in a systemized manner to avoid disruption between any of the files uploaded.

The GitHub centralized repository allows us to steer clear of all the confusion, and therefore, working on the same code becomes effortless and free.

The most important feature of GitHub is “Forking”, copying a repository from one user’s account to another. This helps you to take a project that you do not have right access to and modify it under your account. If users want to make changes, they will need to share it with the original owner by sending a notification called a “Pull request”. Then the owner, with one click of a button, can merge the changes found in the repo with the original repo.

These three features – Fork, pull request, and merge – is what made GitHub more powerful.



Disclaimer: All logos are the property of their respective owners

Understanding the term Git and GitHub

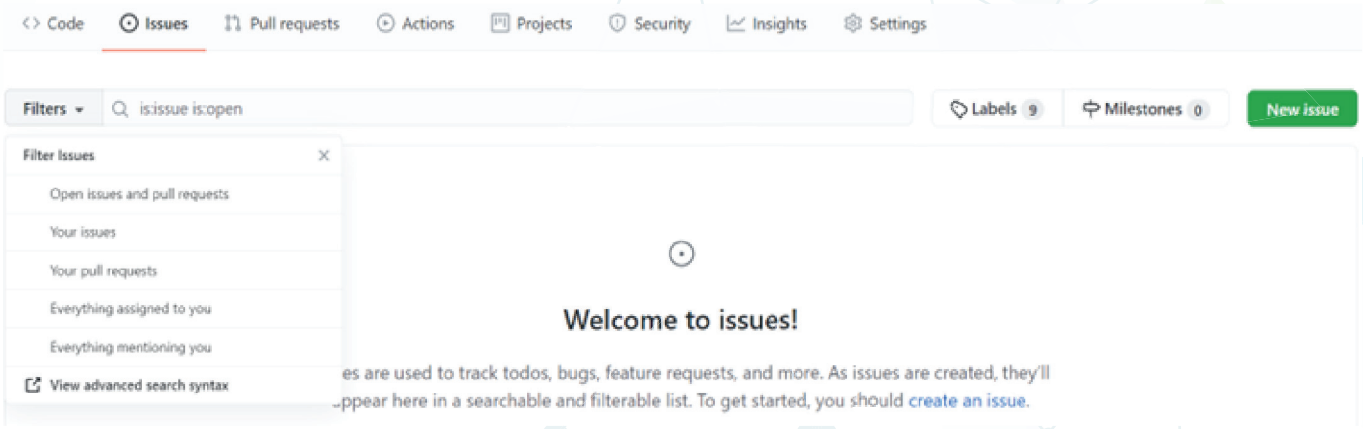
Git is a Distributed Version Control System (DVCS) that helps multiple developers and other contributors work on the same project. It possesses a significant feature to work with one or more local branches and helps them push to a remote repository. Git is responsible for everything GitHub that happens locally on the user's computer. Key features of Git include:

GitHub is a cloud platform that consumes GIT as its core technology. It streamlines the collaboration process and provides a website, command-line tools, and overall flow that allows software developers and users to work together. GitHub is a remote repository.

Features of GitHub

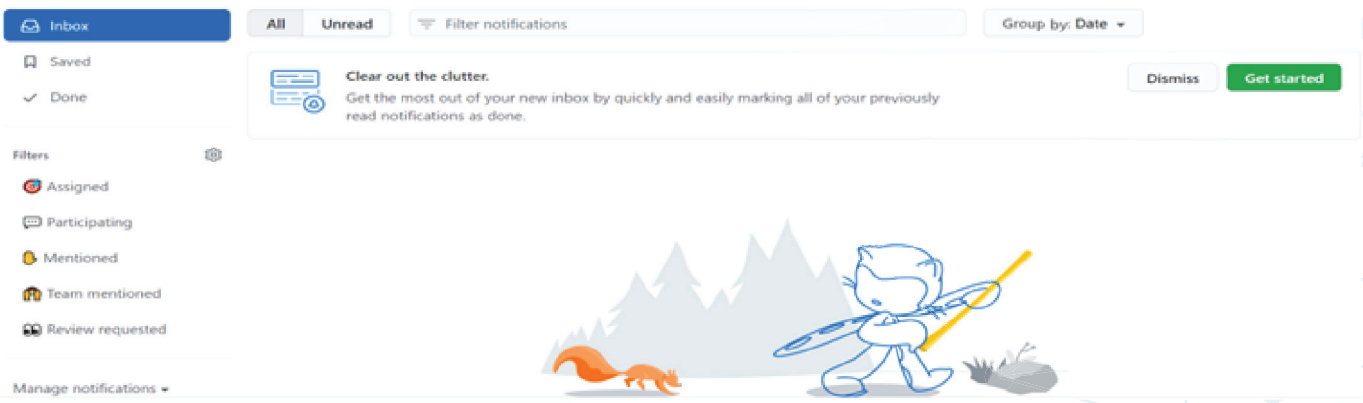
Issues

Issues are where most of the communication between a client and the development team occurs. An issue can be created to discuss a broad set of topics, including bug reports, feature requests, documentation clarifications, etc. Once an issue is created, it can be assigned to owners, labels, projects, and milestones. Developers can also associate issues with pull requests and other GitHub items to provide future traceability.



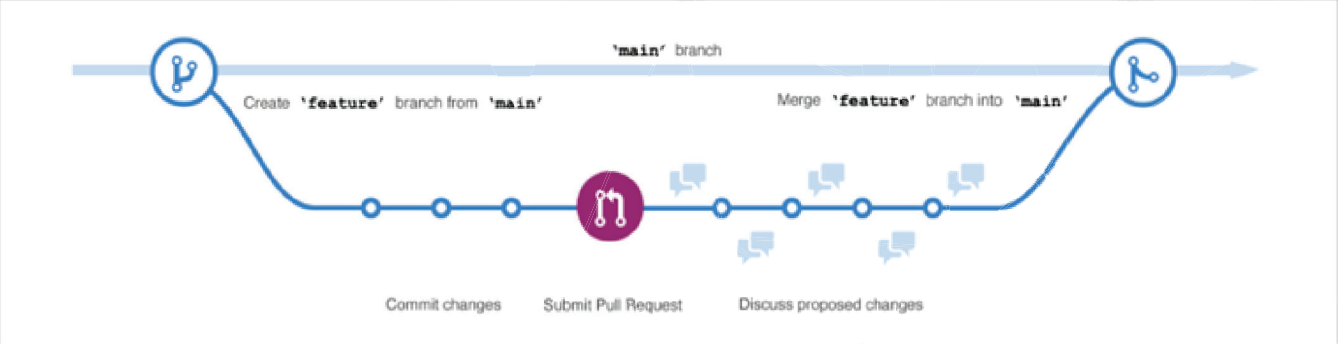
Notifications

GitHub offers notifications for virtually every event that takes place within a workflow. These notifications can be fine-tuned to meet all your preferences. As you can subscribe to all issue creations and edits on a project, or you can just receive notifications for issues that mention you. You can also decide whether you receive notifications via email, web & mobile, or both.



Branches

Branches are the preferred way to create changes in GitHub Flow. They provide isolation so that multiple people may simultaneously work on the same code in a controlled way. This model enables stability among critical branches, such as “main”, while allowing complete freedom for developers to commit any changes they need to meet their goals. Once the code from a branch is ready to become part of the main branch, it may be merged via “pull request”.



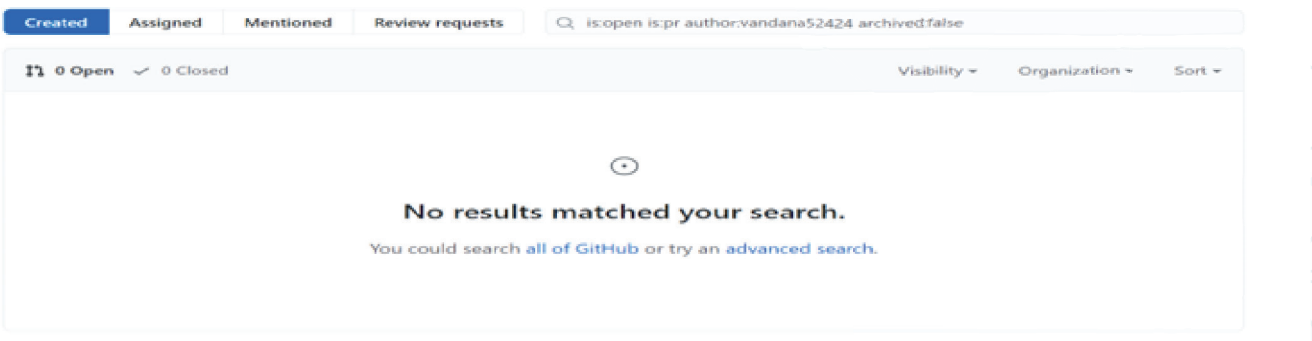
Source: GitHub

Commits

A “commit” is a change to one or more files on a branch. Every time a commit is created, it is assigned a unique ID and tracked along with the time and contributor. This provides a clear audit trail for anyone reviewing the history of a file or linked item, such as an “issue” or “pull request”.

Pull Requests

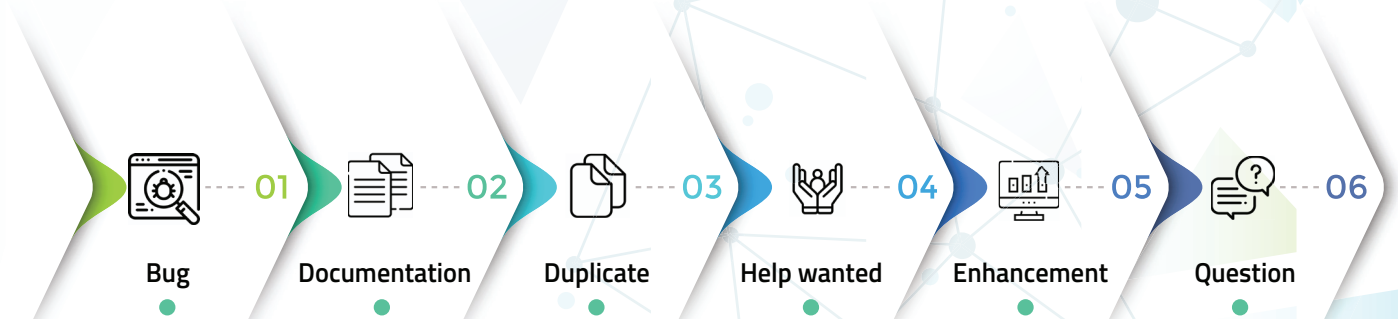
A “pull request” is a way used to signal that the commits from one branch are ready to be merged into another. The developer submitting the pull request will request one or more reviewers to verify the code and approve the merge. These reviewers can comment on changes, add their own, or use the pull request for further discussion. Once the changes have been approved (if approval is required), the pull request's source branch (the compare branch) may be merged into the base branch.



Labels

Labels provide a way to categorize and organize issues and pull requests in a repository. As you create a GitHub repository, several labels will automatically get added for you, and you can even create a new one as well.

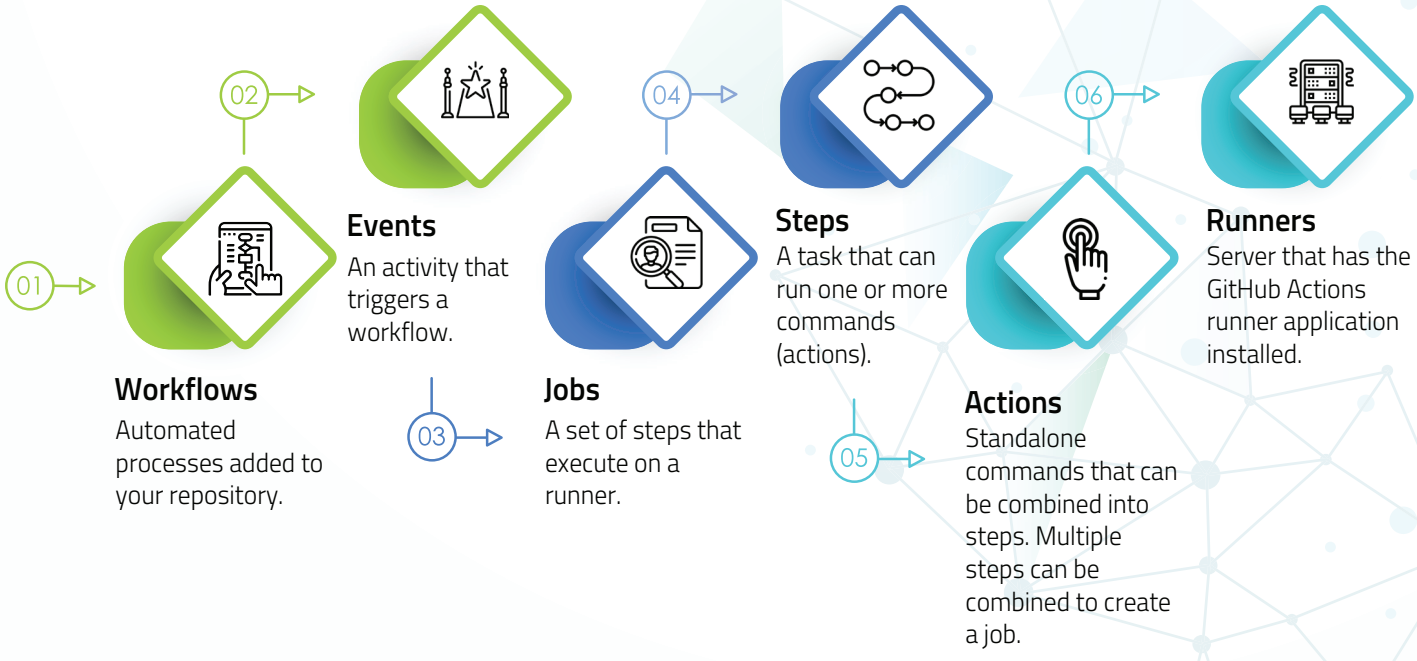
Examples of Labels include:

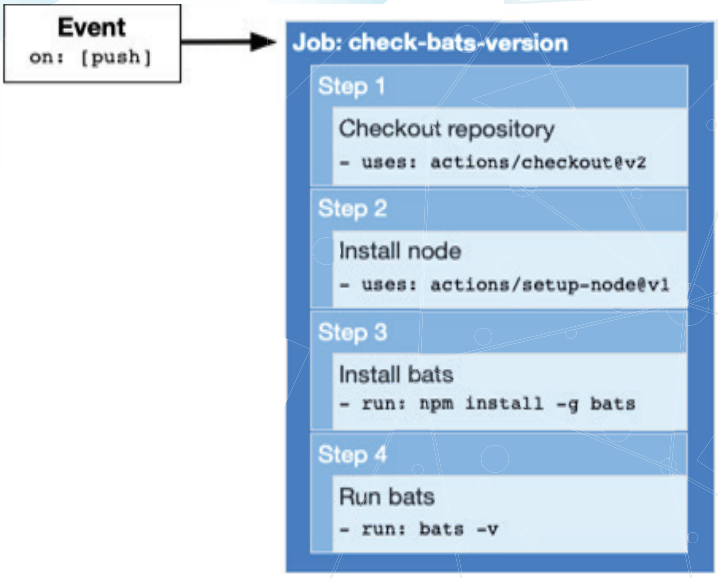


Actions

GitHub actions provide task automation and workflow functionality in a repository. Actions can be used to streamline processes in your software development lifecycle and implement continuous integration and continuous deployment (CI/CD).

GitHub Actions are composed of the following components:





Cloning and Forking

GitHub has vast features to provide multiple ways to copy a repository so users can work on it.

Cloning of a Repository

Cloning a repository will make a duplicate copy of the repository and its history on the user's local machine. If the user has the right access to the repository, they can push changes from their local machine to the remote repository (origin) when it is completed. Users can use the Git clones command or GitHub to clone the repository.

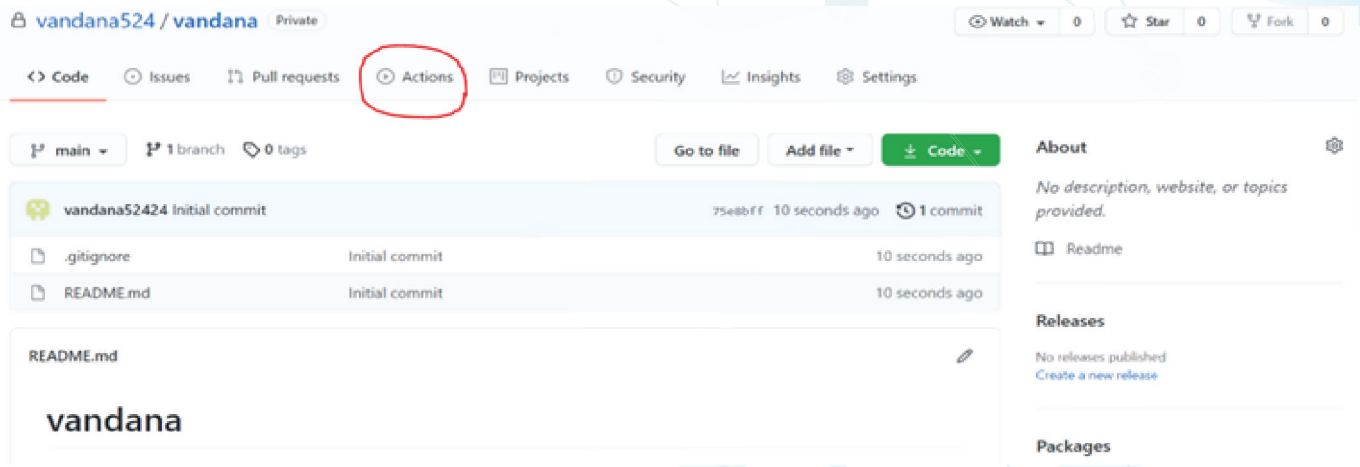
Forking of a Repository

Forking a repository will make a duplicate copy of the repository in the user's GitHub account. Once users have successfully forked a repository into the GitHub account, they can clone it to their local machine. Forking allows you to make changes to a project without affecting the original upstream repository. Users can create a "pull request" to contribute to the upstream repository from your forked repository. Git commands can be used to ensure that the local copy stays synced with the upstream repository.

Build and deploy through GitHub Actions

GitHub Actions can be used to implement continuous integration (CI) for code that is maintained in GitHub repositories. CI is the practice of using automation to build and test software every time a developer commits changes to version control. CI helps teams discover issues early in the development process and fix them quickly.

Several features are under development, and you must make sure the team can build and test them easily so that each one can be quickly added to the website. Because the code for the project is stored in a GitHub repository, you can use GitHub Actions for your CI project.
You can create a workflow from a template — that template has common jobs and steps pre-configured for the automation you are implementing.



Action Logs of Build

Each time the workflow runs, it provides log information indicating whether your workflow deployment was successful. If your workflow is successfully deployed, you will notice a check-mark on the right; if your workflow fails, you will see a red 'X' symbol. Then, you can investigate the reason for those errors/failures and fix them.

Artifacts & Artifact Storage

Workflow produces artifacts along with log entries. This artifact can be uploaded to storage using the action actions/upload-artifact and downloaded from storage using the action actions/download-artifact. Storing an artifact helps to preserve it between jobs. Each job uses a fresh instance of a VM, so you cannot reuse the artifact by saving it on the VM. If you need your artifact in a different job, you can upload the artifact to storage in one job and download it for the other job. Artifacts are stored in storage space on GitHub. The space is free for public repositories, and some amount is free for private repositories depending on the account. GitHub stores your artifact for 90 days.

GitHub Actions to deploy to Azure

Once you have completed the workflow for CI using GitHub Actions and the Artifact is generated, then, with reference to the artifact, you can deploy it to Azure using GitHub features like:

Creating a trigger CD workflow with ChatOps:

ChatOps uses chat clients, chatbots, and real-time communication tools to execute tasks. When you leave a specific comment in a pull request, that can kick off a bot. That bot might comment back with some statistics or run a workflow.

Using labels in your pull request:

Different labels can start different workflows. Add a staging label to begin a deployment workflow to the staging environment or add a spin-up environment label to run the workflow that creates the Microsoft Azure resources you will deploy.

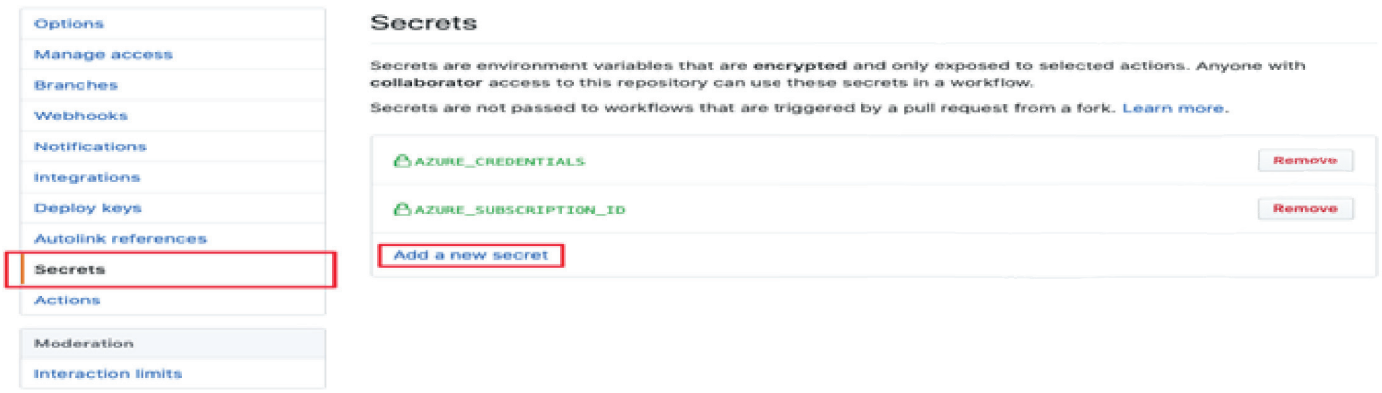
Control Execution

Many developers want to run a workflow if some condition is true. GitHub workflows provide the 'if' conditional for this scenario. The conditional uses an expression that will be evaluated at run time. For example, we want to run this workflow if a staging label is added to the pull request.

GitHub Secrets

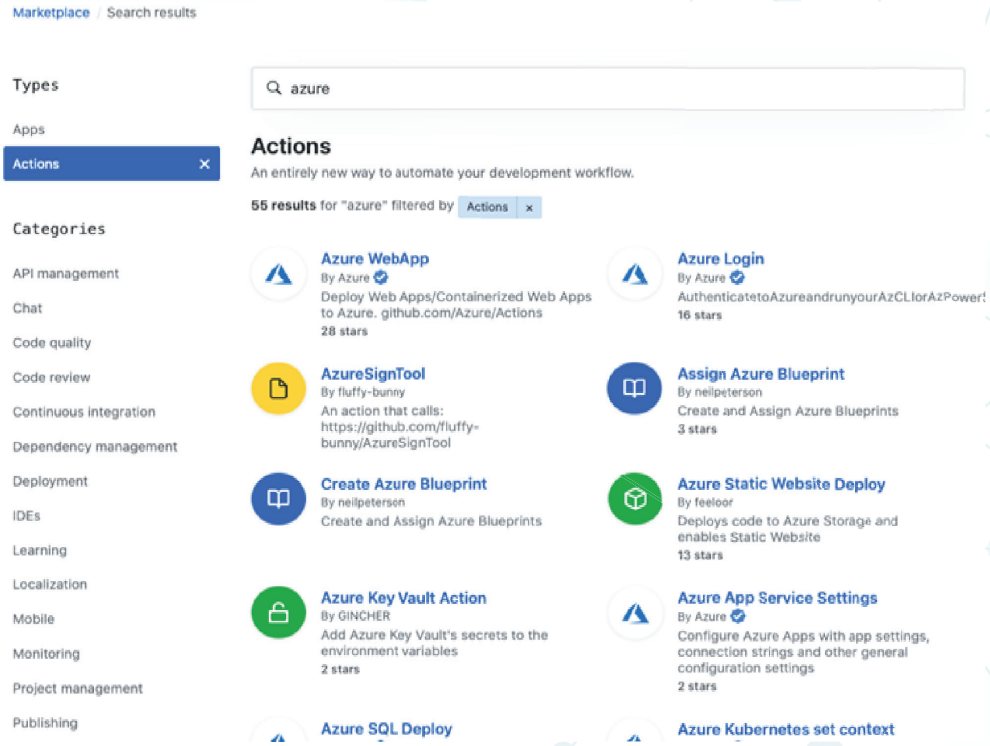
GitHub Secrets is a secure place to store sensitive information that your workflow needs. The GitHub Action must have permission to access the resource to deploy to an Azure resource. You will need to store Azure credentials in plain sight in the workflow file. Instead, you can store your credentials in GitHub Secrets.

To store any information in GitHub Secrets, you can create a "secret" on the portal. You can use the name of the "secret" you created in your workflow wherever you need that information.



Deploy to Microsoft Azure using GitHub Actions

The GitHub Marketplace has several actions that help you automate Azure-related tasks.



You can search and browse GitHub Actions directly in a repository's workflow editor. You can search for a specific Action from the sidebar, view featured Actions, and browse featured categories.

To find an Action:

In your repository, browse to the workflow file that you want to edit.

Click the edit icon in the upper right corner of the file view.

Use the GitHub marketplace sidebar to the right of the editor to browse actions.

Create and delete Azure resources by using GitHub Actions

CD is an automated process when you use the infrastructure as code to create and delete the environment you want to deploy. GitHub Actions can automate these tasks on Azure, and you can include these actions in your workflow. To avoid extra charges, you must delete resources you have created if it is of no use.

One option is to create a new workflow with two jobs, one that spins up resources and deletes them. Then, use a conditional to run only the job you want. In this example, the conditional looks for a label in the “pull request”. It runs the set-up-azure-resources job if the label spins up the environment and the destroy-azure-resources job if the label is destroyed.

Remove workflow Artifacts from GitHub

By default, GitHub provides you 90 days of storage time for your build logs and uploaded artifacts before deletion. This retention period can be customized based on the type of repository and the usage limits set for your specific GitHub product. There is much more information regarding usage limits, artifact retention, billing, and administration. If you are reaching your organization's storage limit for GitHub artifacts and packages, and you want to remove old artifacts without increasing your usage limits and blocking your workflows, you can reclaim used GitHub Actions storage by deleting artifacts before they expire on GitHub

You can do this in the following two ways:

Manually delete Artifacts from your repository

Change the default retention period

Main Benefits of using GitHub

The GitHub Marketplace has several actions that help you automate Azure-related tasks.

01 GitHub makes it easy to contribute to your open-source projects

As we all know, almost every open-source platform uses GitHub for managing their projects. If your project is open source, then using GitHub is free. It includes wiki and issue tracker to make it easy to include more in-depth documentation and feedback about your project. It will be uncomplicated for you to contribute; you will just need to fork a project, make edits as per your requirement, and create a "pull request" to merge using the GitHub web interface.

02 Integration Options

GitHub integrates with common platforms like Amazon and Google cloud, services like Code Climate to track your feedback, and can highlight syntax in 200 different programming languages.

03 GitHub is a repository

GitHub allows you to showcase your projects in front of the public. GitHub is one of the largest coding communities around the world, so it provides wide exposure to your project.

04 Showcase your work

If you are a developer who wants to attract more recruiters to your profile, then GitHub is the best tool you can rely on for this purpose. This is because most companies investigate the GitHub profiles when they search for new talents. So, if you are not from the best-ranked college/university but have a well-versed profile on GitHub, you will have an amazing chance to showcase your talent and get shortlisted and hired by top companies around the world.

05 Documentation

GitHub provides you with excellent exposure to receive great documentation. The help section and guides have articles on topics related to git that you may need and refer to for any project.

06 Markdown

Markdown allows users to use a simple text editor to write formatted documents. GitHub has revolutionized writing by channeling everything through Markdown: from issue tracker, user comments, and much more.

07 Track changes in code across different versions

When multiple developers collaborate on a project, it's hard to keep track of all revisions—who changed what, when, and where those files are stored. GitHub takes care of this issue by keeping track of all the changes that have been pushed to the repository.

Conclusion

Git is an open-source version control system used for source code management in software development and is rapidly gaining ground. Git is not specific to Azure DevOps – a collaborative software development tools set. In fact, it is used by many platforms that provide source control hosting as a service. GitHub's extensive feature set for team-based software development and its ease of use makes it a prominent platform of choice among coders and writers alike. It is also one of the largest communities of coders around. In a nutshell, if you are all in about the community and wish to build and collaborate on open-source projects with millions of diligent developers worldwide, GitHub is just about the right platform for you.

Author Bio

Vandana Yadav has over 5 years of experience in the IT industry. She has vast exposure to Azure Cloud services and has worked with many ventures for their DevOps operations. Vandana is capable of handling CI/CD, infra creation, Monitoring, certificates for security, Application deployments, azure Repo, Git & GitHub, and recently she has worked on Migration services. Before this, Vandana worked with MVP solutions, where she helped clients enhance their infra for cost optimization and efficient services.

happiest minds
The Mindful IT Company
Born Digital . Born Agile

www.happiestminds.com

Business Contact **business@happiestminds.com**

About Happiest Minds Technologies

Happiest Minds Technologies Limited (NSE: HAPPSTMNDS), a Mindful IT Company, enables digital transformation for enterprises and technology providers by delivering seamless customer experiences, business efficiency and actionable insights. We do this by leveraging a spectrum of disruptive technologies such as: artificial intelligence, blockchain, cloud, digital process automation, internet of things, robotics/drones, security, virtual/augmented reality, etc. Positioned as 'Born Digital . Born Agile', our capabilities span digital solutions, infrastructure, product engineering and security. We deliver these services across industry sectors such as automotive, BFSI, consumer packaged goods, e-commerce, edutech, engineering R&D, hi-tech, manufacturing, retail and travel/transportation/hospitality.

A Great Place to Work-Certified™ company, Happiest Minds is headquartered in Bangalore, India with operations in the U.S., UK, Canada, Australia and Middle East.