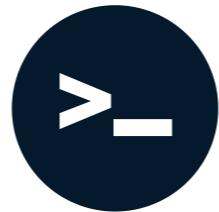


# Intermediate YAML

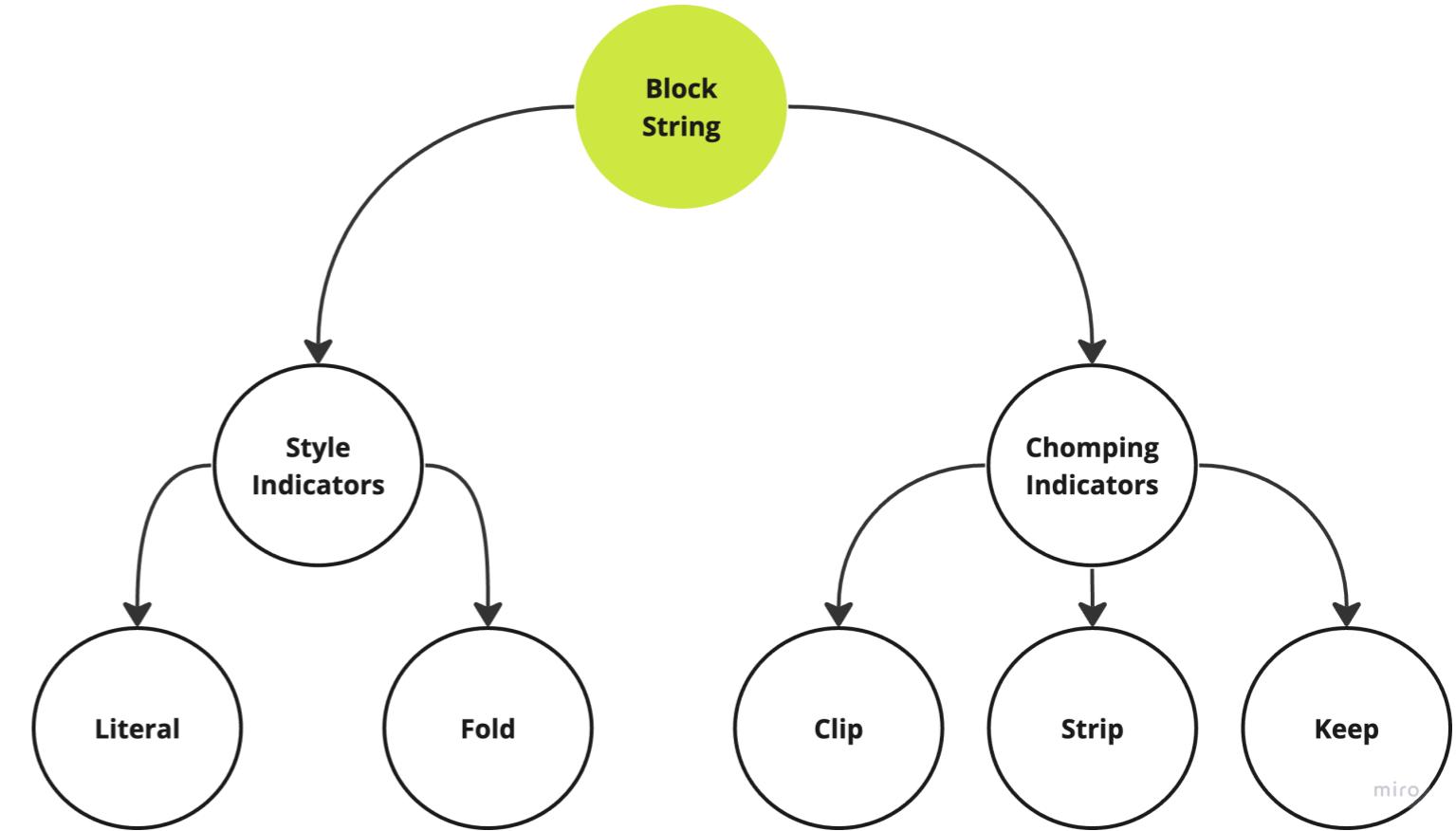
CI/CD FOR MACHINE LEARNING



Ravi Bhadauria  
Machine Learning Engineer

# Multiline strings: Block scalar format

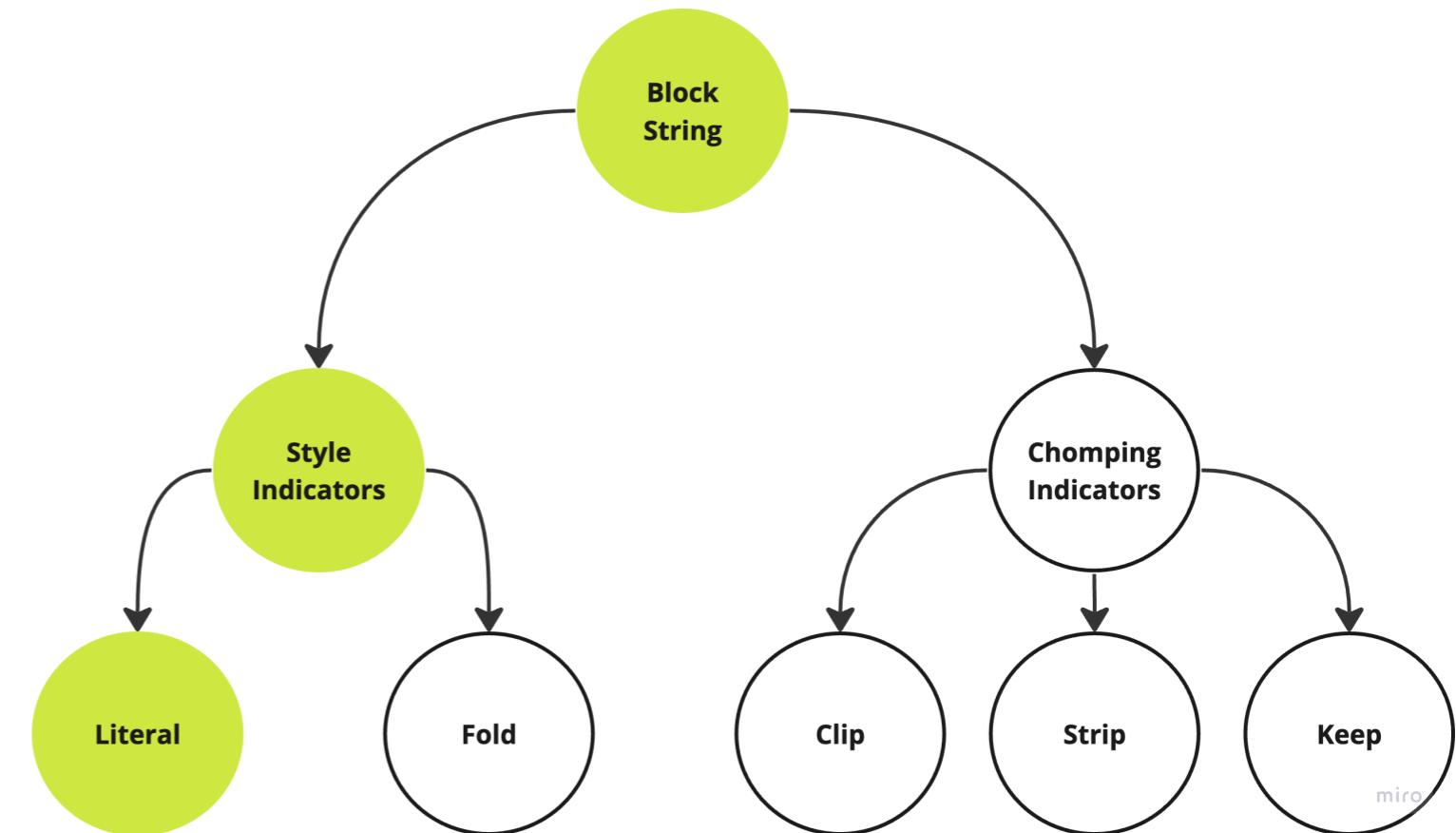
- Represent multiline strings or blocks of text
  - Code snippets (shell commands)
  - Paragraphs (logs)
  - Configuration files



# Style indicators - literal

- Literal style (l): preserves line break and indentation
- Useful for writing shell commands

```
commands: |  
          echo "Running python script..."  
          python3 script.py
```



# Literal style example

YAML

example: I\n

.. Several lines of text,\n.. with some "quotes" of various 'types',\n.. and also a blank line:\n..\n.. and some text with\n.... extra indentation\n.. on the next line,\n.. plus another line at the end.\n..\n..\n

Result

Several lines of text,\nwith some "quotes" of various 'types',\nand also a blank line:\n\nand some text with\nextra indentation\non the next line,\nplus another line at the end.\n

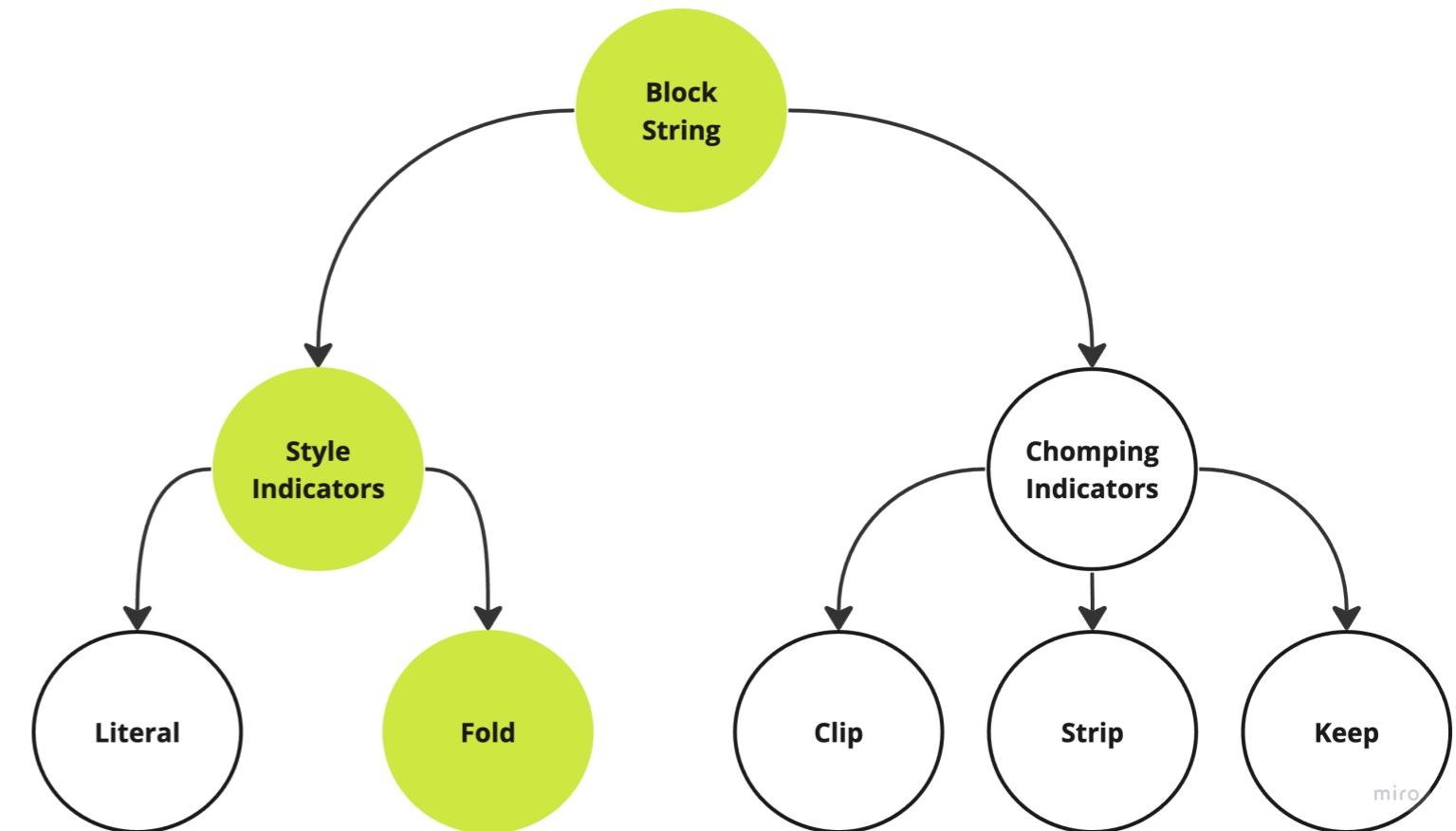
miro

<sup>1</sup> <https://yaml-multiline.info/>

# Style indicators - fold

- Fold style (> ): removes line breaks
- Useful for writing wrapped messages like paragraphs

```
message: >  
Successfully executed the code!  
All processes completed  
without any errors.  
The results have been saved  
in the designated  
output directory.
```



# Fold style example

YAML

example: >\n

- Several lines of text,\n
- with some "quotes" of various 'types',\n
- and also a blank line:\n

... \n

- and some text with\n
- extra indentation\n
- on the next line,\n
- plus another line at the end.\n

...\n

...\n

## Result

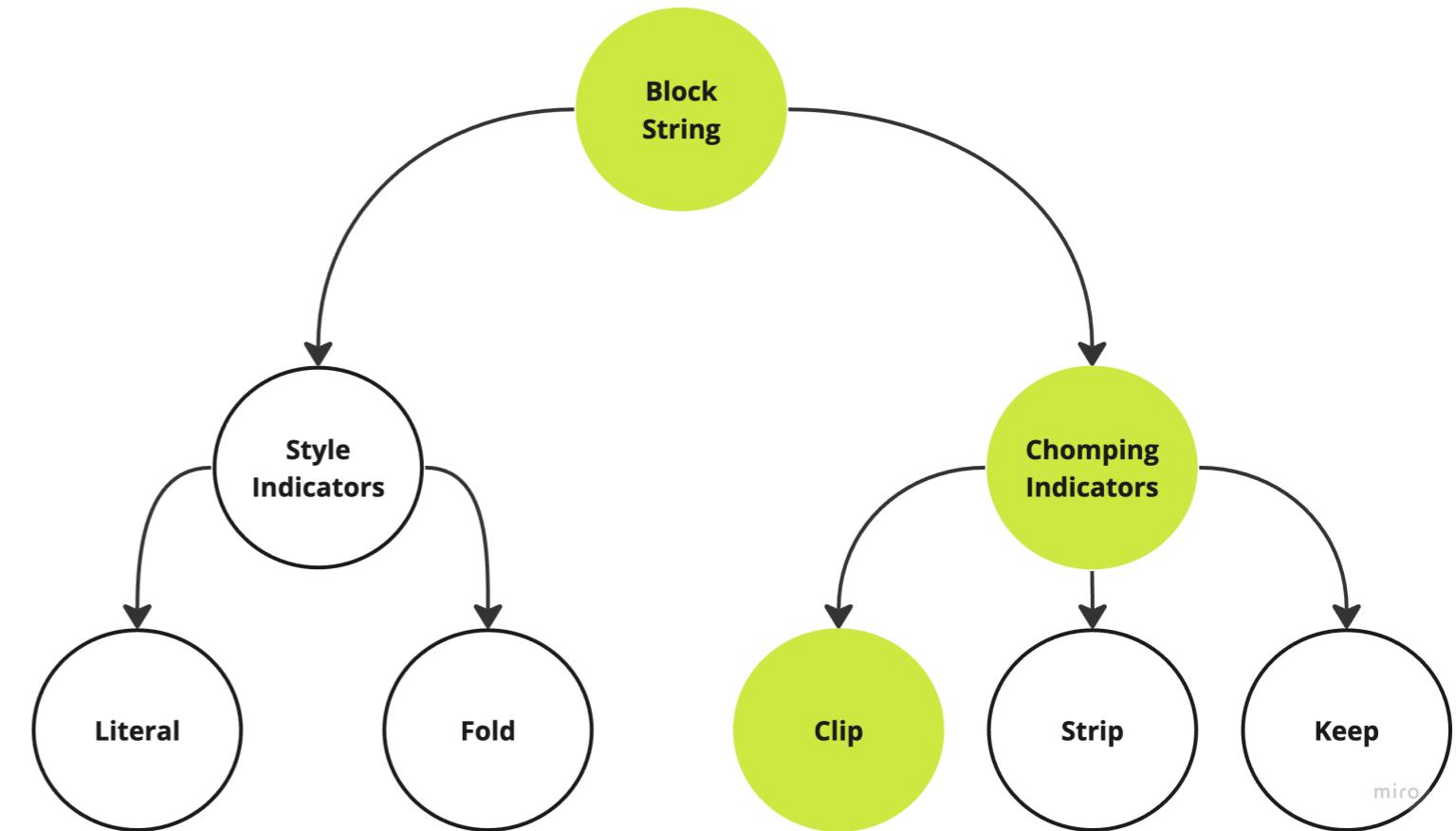
Several lines of text, with some "quotes" of various 'types', and also a blank line:  
and some text with  
extra indentation  
on the next line, plus another line at the end.

miro

<sup>1</sup> <https://yaml-multiline.info/>

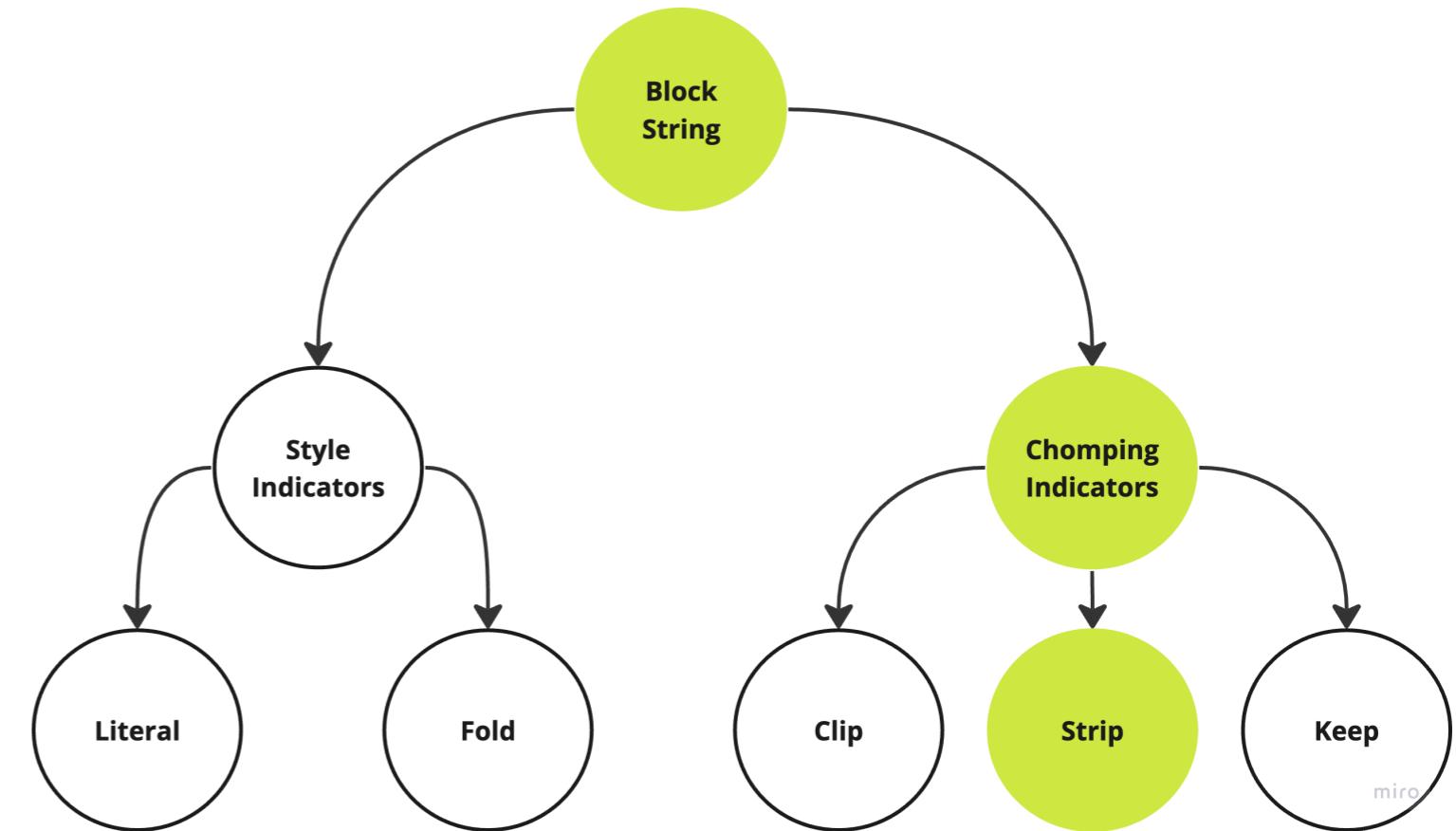
# Chomping indicators

- Control the behavior of newlines at the *end* of the string
- Added after the style indicators
- **clip**: default mode, single newline at the end
  - No additional symbol needed for representation



# Chomping indicators - strip

- `strip (-)`: removes all newlines at the end



# Strip chomping example

YAML

```
example: I-\n
.. Several lines of text,\n
.. with some "quotes" of various 'types',\n
.. and also a blank line:\n
.. \n
.. and some text with\n
.... extra indentation\n
.. on the next line,\n
.. plus another line at the end.\n\n
... \n
... \n
```

Result

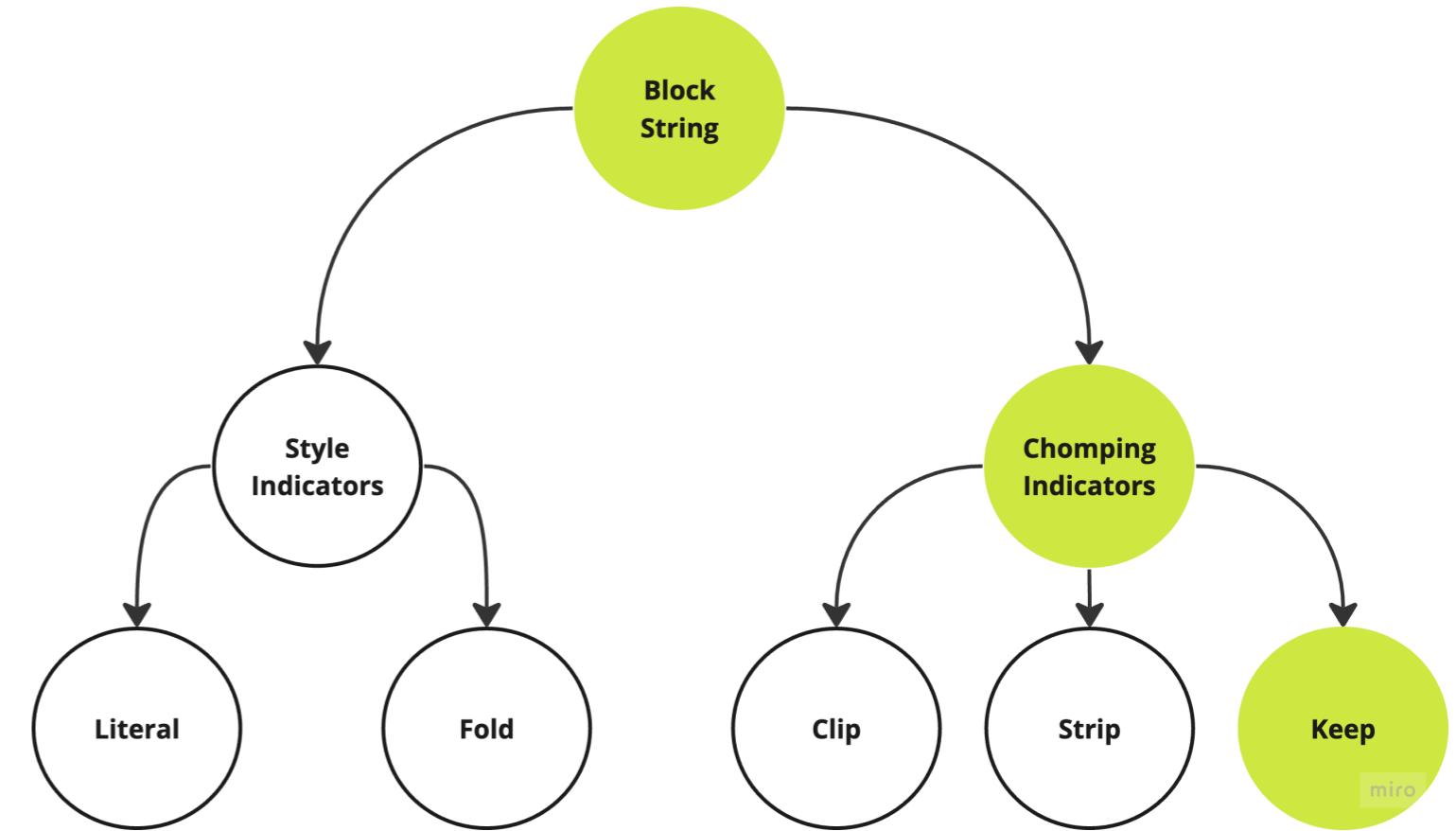
Several lines of text,  
with some "quotes" of various 'types',  
and also a blank line:  
\n  
and some text with  
extra indentation  
on the next line,  
plus another line at the end.

miro

<sup>1</sup> <https://yaml-multiline.info/>

# Chomping indicators - keep

- **keep (+)**: retains all newlines at the end



# Keep chomping example

YAML

example: I+\n

- Several lines of text,\n
- with some "quotes" of various 'types',\n
- and also a blank line:\n
- \n
- and some text with\n
  - extra indentation\n
- on the next line,\n
- plus another line at the end.\n
- \n
- \n

Result

Several lines of text,\nwith some "quotes" of various 'types',\nand also a blank line:\n\nand some text with\nextra indentation\non the next line,\nplus another line at the end.\n\n\n

miro

<sup>1</sup> <https://yaml-multiline.info/>

# Dynamic value injection

- Expressions allow parsers to dynamically substitute values
- Usage:
  - Environment variables
  - References to other parts of YAML
- Typically represented as `${{ ... }}`

```
database:  
  host: ${{ config.database.host }}  
  port: ${{ config.database.port }}  
  username: ${{ config.database.username }}  
  password: ${{ config.database.password }}
```

# Multi-document YAML

- Multiple separate YAML documents within a single file
- Each document contains valid YAML content
- Easy to organize by related documents
- Separate documents by three dashes ---

```
---  
- name: John  
  age: 30  
  
---  
- name: Jane  
  age: 28  
  
---  
- name: Bob  
  age: 35  
  occupation: Developer
```

# Summary

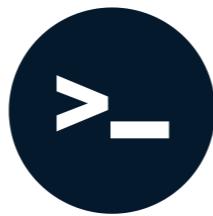
Name	YAML Symbol
Literal Style Indicator	
Fold Style Indicator	>
Strip Chomping Indicator	-
Keep Chomping Indicator	+
Expressions	<code>\${{ ... }}</code>
Multi-document YAML	---

# **Let's practice!**

**CI/CD FOR MACHINE LEARNING**

# Setting a basic CI pipeline

CI/CD FOR MACHINE LEARNING



Ravi Bhadauria  
Machine Learning Engineer

# Anatomy of GitHub Actions workflow

```
# Name of the workflow
name: CI

# Events that trigger workflow
on:
  push:
    branches: [ "main" ]
```

# Anatomy of GitHub Actions workflow

```
# Define jobs and runners
jobs:
  build:
    runs-on: ubuntu-latest
# Define steps
steps:
  - name: Run a multi-line script
    run: |
      echo Hello, world!
      echo Add other actions to build,
      echo test, and deploy your project.
```

# Anatomy of GitHub Actions workflow

```
name: CI

on:
  push:
    branches: [ "main" ]

jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      - name: Run a multi-line script
        run:
          echo Hello, world!
          echo Add other actions to build,
          echo test, and deploy your project.
```

# Create a GitHub repository

- <https://github.com/new>

## Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).

Required fields are marked with an asterisk (\*).

Owner \*



rbhaduria29

Repository name \*

ci-cd-for-ml-demo

ci-cd-for-ml-demo is available.

Great repository names are short and memorable. Need inspiration? How about [jubilant-doodle](#) ?

Description (optional)

Public

Anyone on the internet can see this repository. You choose who can commit.

Private

You choose who can see and commit to this repository.

Initialize this repository with:

Add a README file

This is where you can write a long description for your project. [Learn more about READMEs](#).

Add .gitignore

.gitignore template: Python ▾

Choose which files not to track from a list of templates. [Learn more about ignoring files](#).

Choose a license

License: MIT License ▾

A license tells others what they can and can't do with your code. [Learn more about licenses](#).

This will set main as the default branch. Change the default name in your [settings](#).

You are creating a public repository in your personal account.

[Create repository](#)

# Set up GitHub Actions

- <https://github.com/rbhadauria29/ci-cd-for-ml-demo/settings/actions>
- 

## Actions permissions

### Allow all actions and reusable workflows

Any action or reusable workflow can be used, regardless of who authored it or where it is defined.

### Disable actions

The Actions tab is hidden and no workflows can run.

### Allow rbhadauria29 actions and reusable workflows

Any action or reusable workflow defined in a repository within rbhadauria29 can be used.

### Allow rbhadauria29, and select non-rbhadauria29, actions and reusable workflows

Any action or reusable workflow that matches the specified criteria, plus those defined in a repository within rbhadauria29, can be used. [Learn more about allowing specific actions and reusable workflows to run.](#)

Save

# Setup a "Hello, World!" GitHub Action

The screenshot shows a GitHub repository page for `rbhadauria29 / ci-cd-for-ml-demo`. The `Actions` tab is highlighted with a red box. The repository is public and has 1 branch and 0 tags. The commit history shows an initial commit by `rbhadauria29` 50 minutes ago, which includes files `.gitignore`, `LICENSE`, and `README.md`. The `README.md` file content is displayed as `ci-cd-for-ml-demo`. A watermark for `miro` is visible in the bottom right corner.

rbhadauria29 / ci-cd-for-ml-demo

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

ci-cd-for-ml-demo Public

main 1 branch 0 tags Go to file Add file ▾ ▾ Code ▾

rbhadauria29 Initial commit	57fbeed 50 minutes ago	1 commit
<code>.gitignore</code>	Initial commit	50 minutes ago
<code>LICENSE</code>	Initial commit	50 minutes ago
<code>README.md</code>	Initial commit	50 minutes ago

README.md

ci-cd-for-ml-demo

miro

# Setup a "Hello, World!" GitHub Action

A screenshot of a GitHub repository page for 'rbhadauria29 / ci-cd-for-ml-demo'. The page header shows the repository name and a user profile picture. Below the header, a navigation bar includes links for Code, Issues, Pull requests, Actions (which is highlighted with an orange underline), Projects, Wiki, Security, Insights, and Settings. The main content area features a section titled 'Get started with GitHub Actions' with a sub-section about building, testing, and deploying code. It includes a link to 'set up a workflow yourself →'. Below this is a search bar labeled 'Search workflows'. A 'Suggested for this repository' section displays a card for a 'Simple workflow' by GitHub, which starts with a file having the minimum necessary structure. A red box highlights the 'Configure' button at the bottom of this card.

## Get started with GitHub Actions

Build, test, and deploy your code. Make code reviews, branch management, and issue triaging work the way you want. Select a workflow to get started.

Skip this and [set up a workflow yourself →](#)

Search workflows

### Suggested for this repository

#### Simple workflow

By GitHub

Start with a file with the minimum necessary structure.

[Configure](#)



miro

# Setup a "Hello, World!" GitHub Action

The screenshot shows a GitHub Actions workflow configuration page. On the left, the code editor displays the `hello-world.yml` file for the `ci-cd-for-ml-demo` repository, specifically in the `main` branch. The code defines a basic workflow with a single job named `build` that runs on an Ubuntu runner and echo's "Hello, world!". A red box highlights the entire code block. On the right, a sidebar titled "Marketplace" lists several actions, with a red arrow pointing from the "Commit changes..." button at the top right towards the "Close Stale Issues" action.

```
1 # This is a basic workflow to help you get started with Actions
2
3 # Name of the workflow |
4 name: CI
5
6 # Controls when the workflow will run
7 on:
8   # Triggers the workflow on push events but only for the "main" branch
9   push:
10     branches: [ "main" ]
11
12 # A workflow run is made up of one or more jobs that can run sequentially or in parallel
13 jobs:
14   # This workflow contains a single job called "build"
15   build:
16     # The type of runner that the job will run on
17     runs-on: ubuntu-latest
18
19   # Steps represent a sequence of tasks that will be executed as part of the job
20   steps:
21
22     # Runs a set of commands using the runners shell
23     - name: Run a multi-line script
24       run:
25         echo Hello, world!
26         echo Add other actions to build,
27         echo test, and deploy your project.
```

Marketplace Documentation

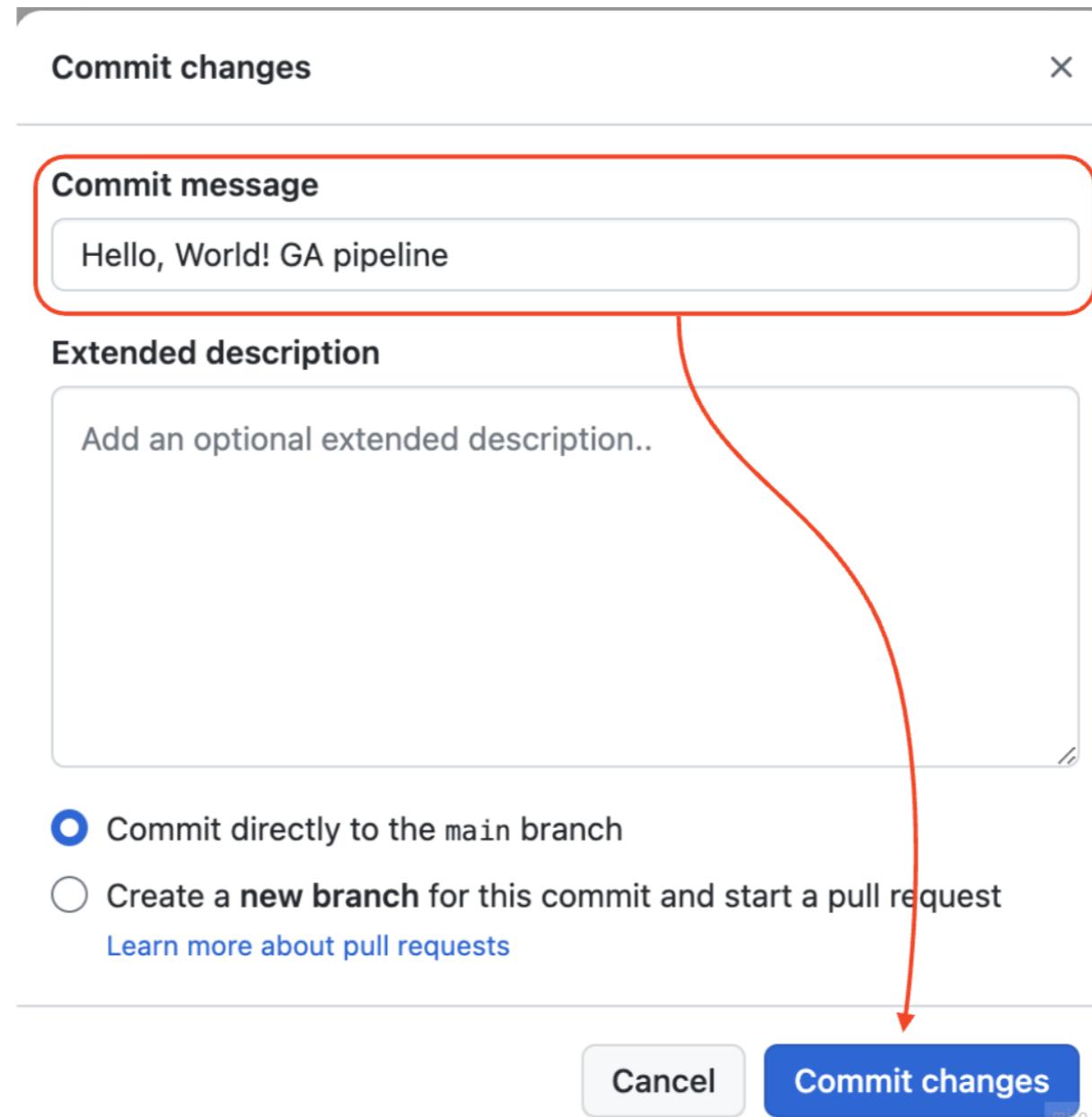
Search Marketplace for Actions

Featured Actions

- Setup Node.js environment** By actions 3.1k  
Setup a Node.js environment by adding problem matchers and optionally downloading and adding it to the PATH
- Setup Java JDK** By actions 1.2k  
Set up a specific version of the Java JDK and add the command-line tools to the PATH
- Setup Go environment** By actions 1.1k  
Setup a Go environment and add it to the PATH
- Close Stale Issues** By actions 1k  
Close issues and pull requests with no recent activity
- First interaction** By actions 633  
Greet new contributors when they create their first issue or open their first pull request

Cancel changes Commit changes...

# Setup a "Hello, World!" GitHub Action



# Inspect the GitHub Actions Run

	rbhadauria29	Hello, World! GA pipeline	 e49adbd 12 minutes ago	 5 commits
	.github/workflows	Hello, World! GA pipeline		12 minutes ago
	.gitignore	Initial commit		yesterday
	LICENSE	Initial commit		yesterday
	README.md	Initial commit		yesterday

# Inspect the GitHub Actions Run

All workflows

Showing runs from all workflows

Filter workflow runs

1 workflow run

Event ▾

Status ▾

Branch ▾

Actor ▾

✓ Hello, World! GA pipeline

CI #4: Commit e49adbd pushed by rbhadauria29

main

11 minutes ago

12s

...

miro

Triggered via push 1 hour ago

rbhadauria29 pushed → e49adbd main

Status

Success

Total duration

12s

Artifacts

—

hello-world.yml

on: push

✓ build

1s



# Inspecting the Logs

← CI

✓ Hello, World! GA pipeline #4

Re-run all jobs ⋮

Summary

Jobs

build

succeeded 1 hour ago in 1s

Search logs

Run details

Usage

Workflow file

build

Set up job

- 1 Current runner version: '2.307.1'
- 2 ► Operating System
- 6 ► Runner Image
- 11 ► Runner Image Provisioner
- 13 ► GITHUB\_TOKEN Permissions
- 17 Secret source: Actions
- 18 Prepare workflow directory
- 19 Prepare all required actions
- 20 Complete job name: build

Run a multi-line script

- 1 ► Run echo Hello, world!
- 6 Hello, world!
- 7 Add other actions to build,
- 8 test, and deploy your project.

Complete job

- 1 Cleaning up orphan processes

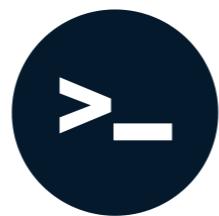
The screenshot shows a CI pipeline interface with a dark theme. On the left, there's a sidebar with links like 'Summary', 'Jobs', 'Run details', 'Usage', and 'Workflow file'. The main area displays a single job named 'build' which succeeded 1 hour ago. The job has three steps: 'Set up job', 'Run a multi-line script', and 'Complete job'. The 'Run a multi-line script' step is highlighted with a red rounded rectangle. The log output for this step shows the command 'echo Hello, world!' being run and the resulting output 'Hello, world!'. The 'Complete job' step shows the final action of cleaning up orphan processes.

# **Let's practice!**

**CI/CD FOR MACHINE LEARNING**

# Running repository code

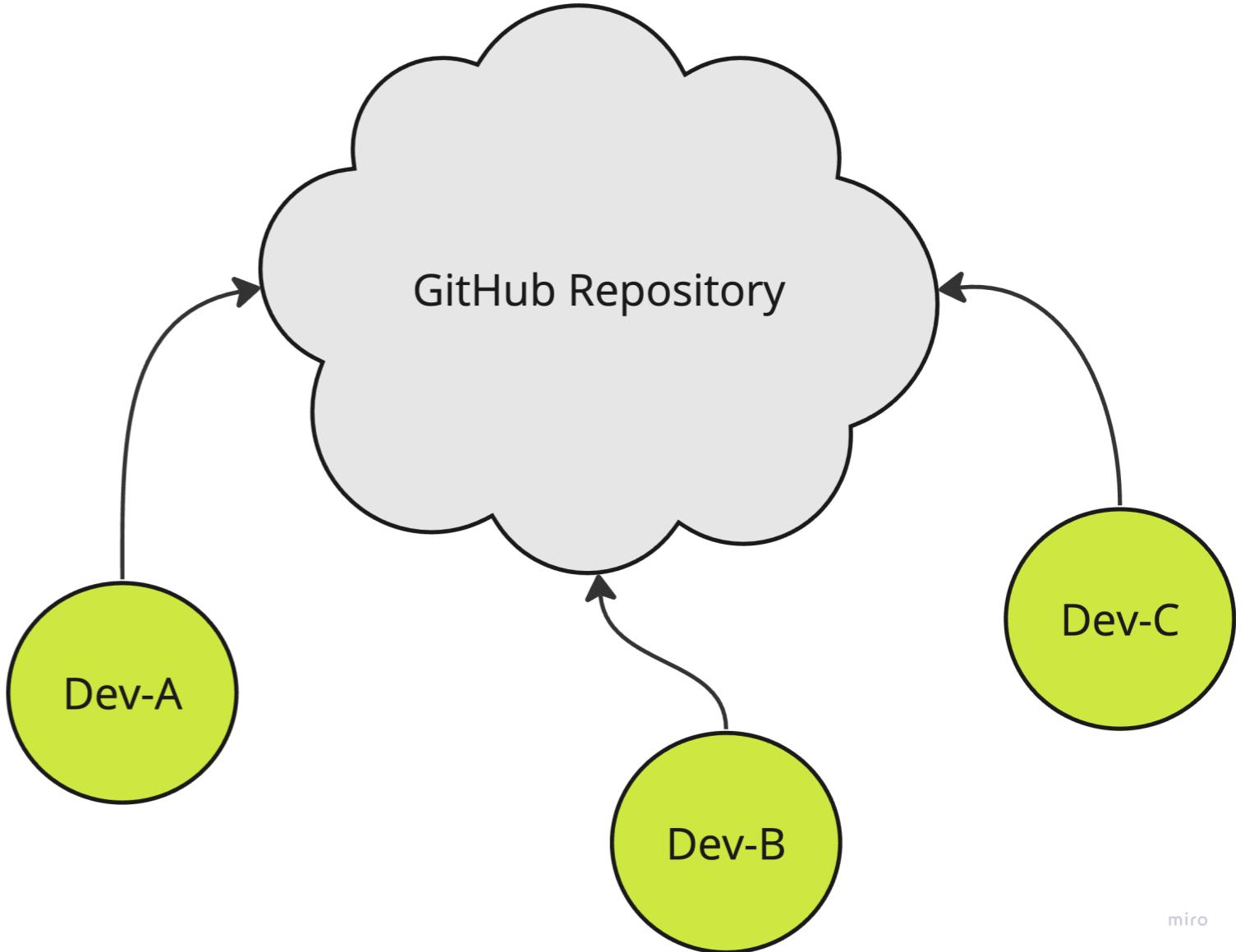
CI/CD FOR MACHINE LEARNING



Ravi Bhadauria  
Machine Learning Engineer

# Shared repository model

- Multiple collaborators work on same repository
- Feature/Topic branches are created for related work
  - Pull requests are opened
  - Code is reviewed and changes are suggested
  - Branch code is merged into `main` branch
- CI/CD enabled checks improve code quality



miro

# Create a feature branch

The screenshot shows a GitHub repository page for a public repository named "ci-cd-for-ml-demo". The repository has 1 branch and 0 tags. The "Code" tab is selected. A search bar is present, and a navigation bar includes tabs for Overview, Yours, Active, Stale, All branches, and New branch. The "New branch" button is highlighted with a red box. Below the navigation bar, a "Default branch" section shows the "main" branch, which was updated 9 hours ago by rbhaduria29. The "Default" status is indicated by a checkmark and a "Default" badge.

ci-cd-for-ml-demo Public

Pin Unwatch 1 Fork 0 Star 0

main 1 branch 0 tags Go to file Add file Code About

Search branches... Overview Yours Active Stale All branches New branch

Default branch

main Updated 9 hours ago by rbhaduria29 ✓ Default ↵ miro

# Create a feature branch



ci-cd-for-ml-demo

Public

Pin

Unwatch 1

Fork 0

Star 0

pr-workflow ▾

2 branches

0 tags

Go to file

Add file ▾

Code ▾

About



No description, website, or topics provided.

miro

This branch is up to date with main.

Contribute ▾

Readme

# Add repository code

Save as `hello_world.py` in the branch

```
import datetime

def main():
    print("Hello, World!")

    # Get the current time and print it
    current_time = datetime.datetime.now()
    print("Current time:", current_time)

if __name__ == "__main__":
    main()
```

# Configure workflow event

- Specify target branch for pull request event

```
on:  
  pull_request:  
    branches: [ "main" ]
```

# Actions syntax

steps:

```
- name: My Action  
  uses: <org_or_user_name>/<reponame>@<version_number>
```

with:

```
  action-argument: value
```

- Action repository [https://github.com/<org\\_or\\_user\\_name>/<reponame>](https://github.com/<org_or_user_name>/<reponame>)
  - E.g. <https://github.com/actions/checkout>
- Arguments can be provided using *with* key
- GitHub marketplace actions at <https://github.com/actions>

# Configure workflow steps and actions

- Checking out repository in job step
- Setting up python environment

```
steps:
```

- name: Checkout
- uses: actions/checkout@v3

```
steps:
```

- name: Setup Python
  - uses: actions/setup-python@v4
- ```
with:
```
- ```
python-version: 3.9
```

# Putting it together

- Workflow triggers on pull request to `main`
- `build` job contains three steps
  - `Checkout` checks out the repository
  - `Setup Python` sets up the Python environment
  - `Run Python script` runs the Python file

```
name: PR

on:
  pull_request:
    branches: [ "main" ]

jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      - name: Checkout
        uses: actions/checkout@v3
      - name: Setup Python
        uses: actions/setup-python@v4
        with:
          python-version: 3.9
      - name: Run Python script
        run:
          echo hello_world.py
          python hello_world.py
```

# Create PR and trigger workflow

testing PR workflow #1

[Open](#) rbhaduria29 wants to merge 2 commits into `main` from `pr-workflow`

Conversation 0 Commits 2 Checks 0 Files changed 2 +46 -0

rbhaduria29 commented now  
No description provided.

rbhaduria29 added 2 commits 47 minutes ago

- o Create `hello_world.py` Verified b93182b
- o Create `pr.yml` Verified bcc63f6

Add more commits by pushing to the `pr-workflow` branch on [rbhaduria29/ci-cd-for-ml-demo](#).

Some checks haven't completed yet Hide all checks

- PR / build (pull\_request) Queued — Waiting to run this check... Details
- This branch has no conflicts with the base branch Merging can be performed automatically.

Merge pull request You can also [open this in GitHub Desktop](#) or view [command line instructions](#).

Reviewers No reviews Still in progress? Convert to draft

Assignees No one—assign yourself

Labels None yet

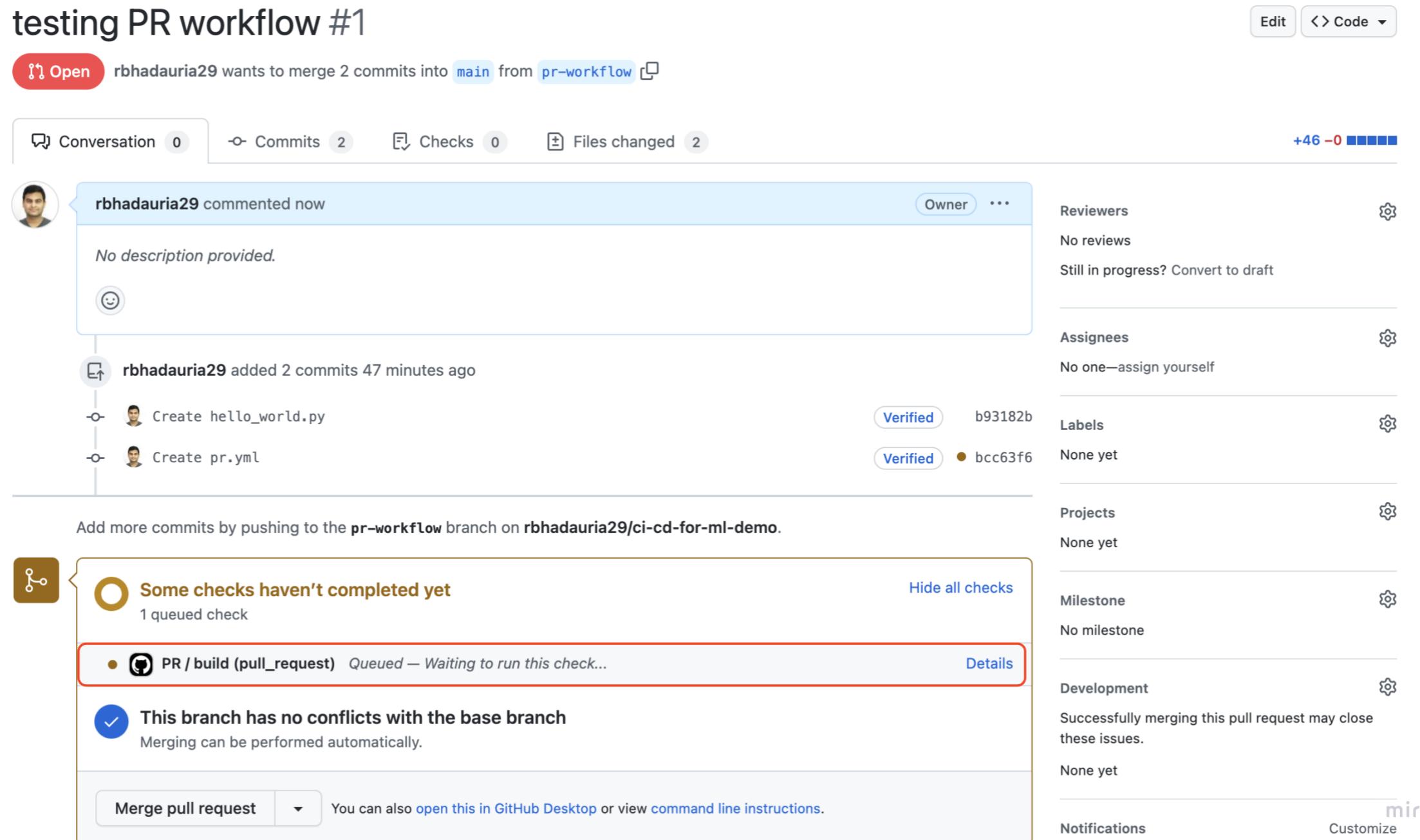
Projects None yet

Milestone No milestone

Development Successfully merging this pull request may close these issues.

None yet

Notifications Customize



# Inspect workflow logs

The screenshot shows a GitHub Actions workflow log for a pull request. The top navigation bar includes links for Code, Issues, Pull requests (1), Actions (highlighted), Projects, Wiki, Security, Insights, and Settings. The main title is "testing PR workflow #1". On the left, a sidebar lists Summary, Jobs (with "build" selected), Run details, Usage, and Workflow file. The main content area displays the "build" job, which succeeded 16 hours ago in 5s. The log details the steps: Set up job (2s), Checkout (1s), Setup Python (0s), Run Python script (0s). The "Run Python script" step shows the command "Run echo hello\_world.py" and its output: "hello\_world.py", "Hello, World!", and "Current time: 2023-08-04 01:11:24.991641". Subsequent steps include Post Setup Python, Post Checkout, and Complete job, all completed in 0s.

```
build
succeeded 16 hours ago in 5s

> ✓ Set up job 2s
> ✓ Checkout 1s
> ✓ Setup Python 0s
✓ Run Python script 0s
  1 ► Run echo hello_world.py
  12 hello_world.py
  13 Hello, World!
  14 Current time: 2023-08-04 01:11:24.991641

> ✓ Post Setup Python 0s
> ✓ Post Checkout 0s
> ✓ Complete job 0s
```

# **Let's practice!**

**CI/CD FOR MACHINE LEARNING**

# Environment Variables and Secrets

CI/CD FOR MACHINE LEARNING



Ravi Bhadauria  
Machine Learning Engineer

# Contexts

- Access information about predefined variables and data
  - workflow runs
  - variables
  - runner environments
  - jobs and steps
- Access contexts using the expression syntax `${{ context.XXX }}`
- Contexts used in this course
  - `github` : information about the workflow run
  - `env` : variables set in the workflow
  - `secrets` : names and values that are available to workflow
  - `job` : info about the current job
  - `runner` : info about the machine

<sup>1</sup> <https://docs.github.com/en/actions/learn-github-actions/context>

# Variables

- Store non-sensitive information in plain text
  - compiler flags, usernames, file paths
- Declared as value for `env` key
- Global/local scope is controlled by the level where defined
- Accessed from the `env` context as `${{ env.ENV_VAR }}`

```
name: Greeting on variable day
# Global env

env:
  Greeting: Hello

jobs:
  greeting_job:
    runs-on: ubuntu-latest
    # Local env: scoped to greeting_job
    env:
      First_Name: Ravi

    steps:
      - run:
          echo "${{ env.Greeting }} \
${{ env.First_Name }}."
```

# Secrets

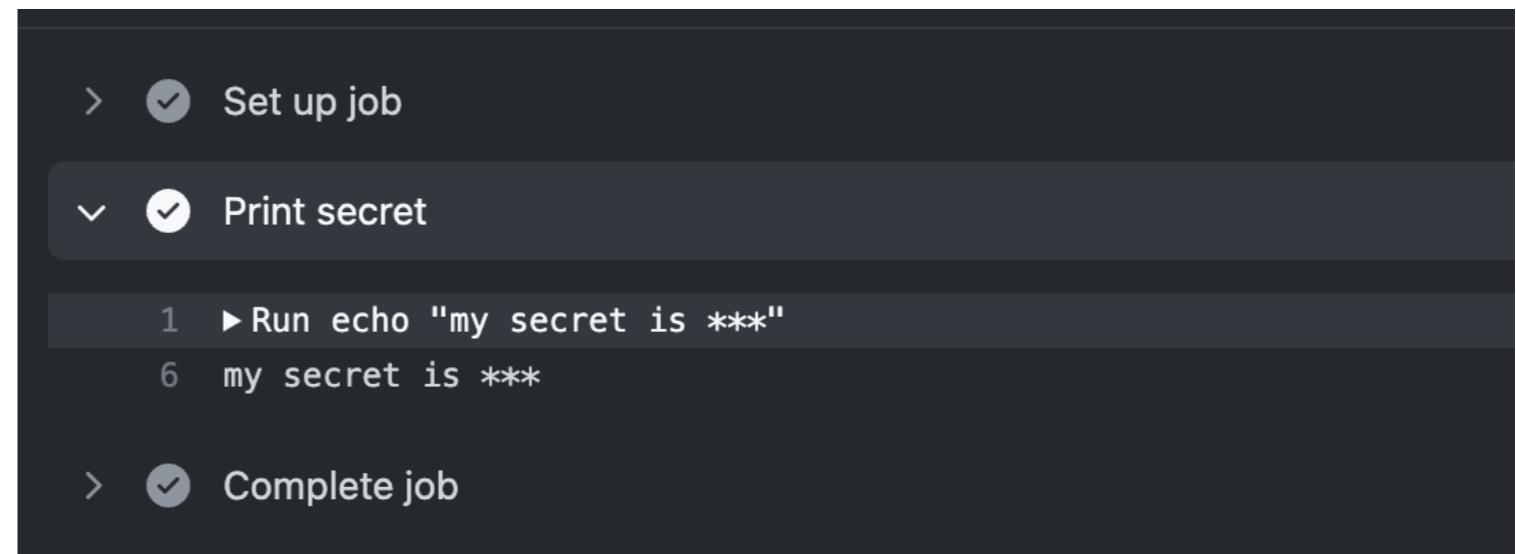
- Store sensitive information in encrypted manner
  - passwords, API keys
- Access values via `secrets` context
  - `${{ secrets.SuperSecret }}`
- Can store them as input or environment variable

```
steps:  
  - name: Hello world action  
    env: # Set the secret as an env var  
      super_secret: ${{ secrets.SuperSecret }}  
    with: # Or as an input  
      super_secret: ${{ secrets.SuperSecret }}
```

- Printing secret

```
steps:  
  - name: Print secret  
    run: |  
      echo "my secret is \  
      ${{ secrets.SuperSecret }}"
```

- Output



The screenshot shows a dark-themed CI/CD pipeline interface. It displays a list of steps:

- >  Set up job
- ▼  Print secret
  - 1 ► Run echo "my secret is \*\*\*"
  - 6 my secret is \*\*\*
- >  Complete job

The 'Print secret' step is expanded, showing the command run (echo "my secret is \*\*\*") and its output (my secret is \*\*\*). The first line of the output is preceded by a green checkmark icon.

# Setting secrets

A screenshot of a GitHub repository page for 'rbhadauria29 / ci-cd-for-ml-demo'. The page includes a navigation bar with links for Code, Issues, Pull requests (1), Actions, Projects, Wiki, Security, Insights, and Settings. The Settings link is highlighted with a red box and an arrow pointing to it from the list below.

- Go to **Security > Secrets and Variables > Actions**

A screenshot of the 'Actions secrets and variables' page. It features a green button labeled 'New repository secret' with a red box and an arrow pointing to it from the list above. Below the button, there is descriptive text about secrets and variables. At the bottom left, there are two tabs: 'Secrets' (which is highlighted with a red box and an arrow) and 'Variables'. The 'miro' logo is visible at the bottom right.

Actions secrets and variables

New repository secret

Secrets and variables allow you to manage reusable configuration data. Secrets are **encrypted** and are used for sensitive data. [Learn more about encrypted secrets](#). Variables are shown as plain text and are used for **non-sensitive** data. [Learn more about variables](#).

Anyone with collaborator access to this repository can use these secrets and variables for actions. They are not passed to workflows that are triggered by a pull request from a fork.

Secrets Variables miro

# Setting secrets

## Actions secrets / New secret

Name \*

Secret \*

Add secret

# GITHUB\_TOKEN secret

- Built in secret provided by GitHub Actions
- Used to perform workflow actions
  - Cloning the repository and fetching code
  - Opening and closing issues and pull requests
  - Commenting on issues and pull requests
- Automatically available in every GitHub Actions workflow
  - Accessed via `$${{ secrets.GITHUB_TOKEN }}`
- Permissions can be tuned to the right degree

# Example: commenting on a pull request

- Grant permissions to write comments in PR

```
permissions:
```

```
  pull-requests: write
```

- Use GITHUB\_TOKEN to authorize

```
permissions:
```

```
  pull-requests: write
```

```
steps:
```

```
  - name: Comment PR
```

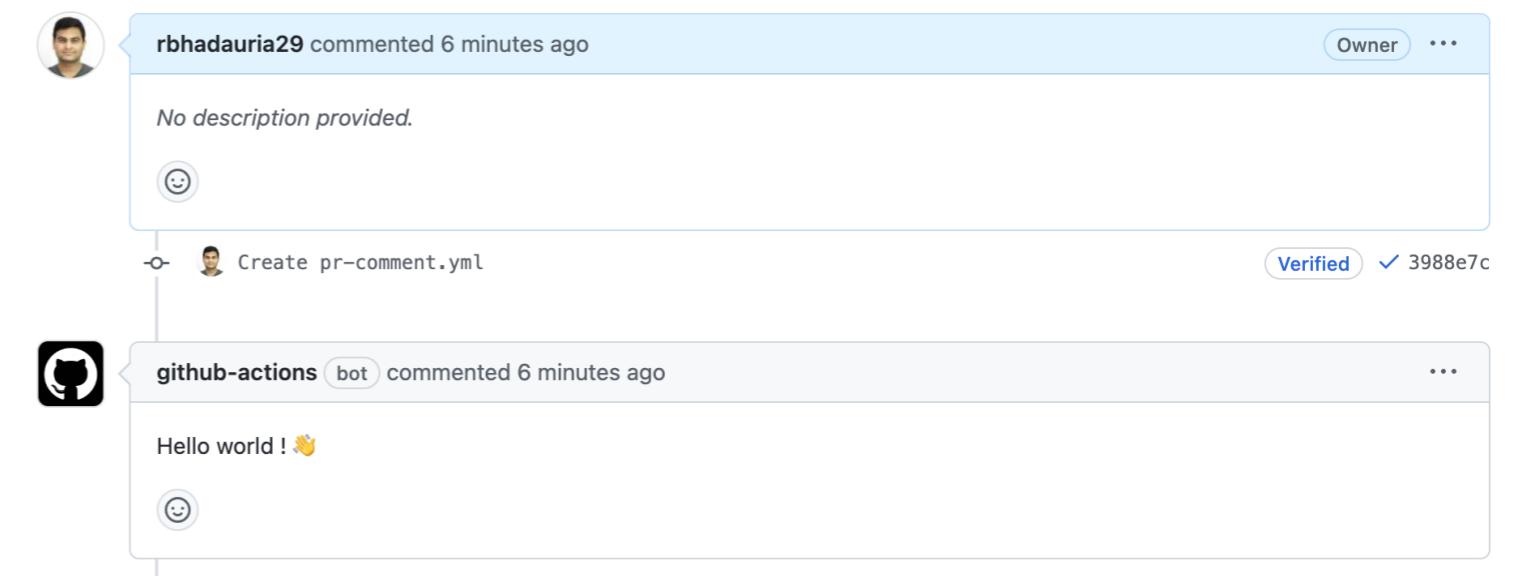
```
    uses: thollander/actions-comment-pull-request@v2
```

```
    with:
```

```
      GITHUB_TOKEN: ${{ secrets.GITHUB_TOKEN }}
```

```
      message: |
```

```
        Hello world ! :wave:
```



<sup>1</sup> <https://gist.github.com/rbhadauria29/6d7fc51944b4fb48425c3c307fec77c6>

# **Let's practice!**

**CI/CD FOR MACHINE LEARNING**