# Git & Github Workshop

Looking Under the Hood

# Folder vs. Directory vs. Repository

Folder

Folder on your computer to store files under a single name.

More user friendly name for a directory.

Directory

Folder on your computer to store files under a single name.

Stems from the organizational system of the filesystem, allowing files to be more easily findable

Repository

git workspace

Where a git project is located, can be identified because the root of the repo contains a .git folder where all the git data is stored.

## Folder = Directory != Repository

# Git & Github Differences

| Git | Github |
| --- | --- |
| Runs on your local machine | Public facing UI to manage your repo |
| Version Control | Remote Repository |
| Workspace | Portfolio |

# Git - Looking Under the Hood: Basic Commands

git init

git clone <remote-repo-url>

git add <file-name> or git add .

git commit -m “Commit Message Required”

git push

git pull

# Git - Looking Under the Hood: Basic Workflow

Create Git Repository

Create and develop code

Stage Code

Commit Code

Push Code

Pull Code

# Git - Looking Under the Hood: Basic Workflow

Create Git Repository

Create and develop code

This is the part where you connect any regular folder on your computer to git.

Use: ls -la on bash or ls on powershell

Initializing a starting point for your version control history as you develop your code.

Initializing your code’s connection to the world wide web through remote repos

# Local Repo vs. Remote Repo

## After making changes to a file

### Local Repo

A local repository is on that exists on the machine you are currently working on.

A local repo can be made using either git init or if it already exists you can get it on your local machine using git clone <remote-repo-url>

### Remote Repo

A remote repository is one that exists hosted on a server; ie. github, bitbucket or gitlab.

# Staged vs. Unstaged

## After making changes to a file

### Unstaged

Before applying

git add . or git add <file–name>

Those changes are unstaged; won’t be part of your next commit

Use .gitignore to specify files or folders to never add. (note if the file has already been committed then adding it to the .gitignore file will not stop tracking it).

### Staged

After applying

git add . or git add <file-name>

Those changes become staged; will be part of your next commit.

# Git - Looking Under the Hood: Basic Workflow

Commit Code

Think of this as creating a record for the set of changes you've added to your git repository.

(ls -la or dir /a or ls -Force to look at hidden .git folder)

When using commit, your saving this record locally on your machine. This record is what's used to reference every update you've made when developing your code over time.

(.git folder holds the record of all your commits)

You should try to **only commit working code** whenever possible.

Us the command below to look at your history of changes

git log

# Git - Looking Under the Hood: Basic Workflow

## Push Code

This is where you send all the commits you've made to a remote repository, like github, bitbucket or gitlab.

Now you can access those changes from any computer with an internet connection and git to continue developing from anywhere in the world.

Now you can collaborate with other developers on a single project.

Now you can showcase your work to the rest of the world, and give them a picture of where you currently find yourself as a developer and the path your taking.

# Git - Looking Under the Hood: Basic Workflow

## Pull Code

With this command you can now pull any changes you've made to your git repository from anywhere you've established a connection to it.

# Git - Looking Under the Hood: Basic Commands

## git init or git clone <remote-repo-url>

### git init

Starts a git repository locally on your machine.

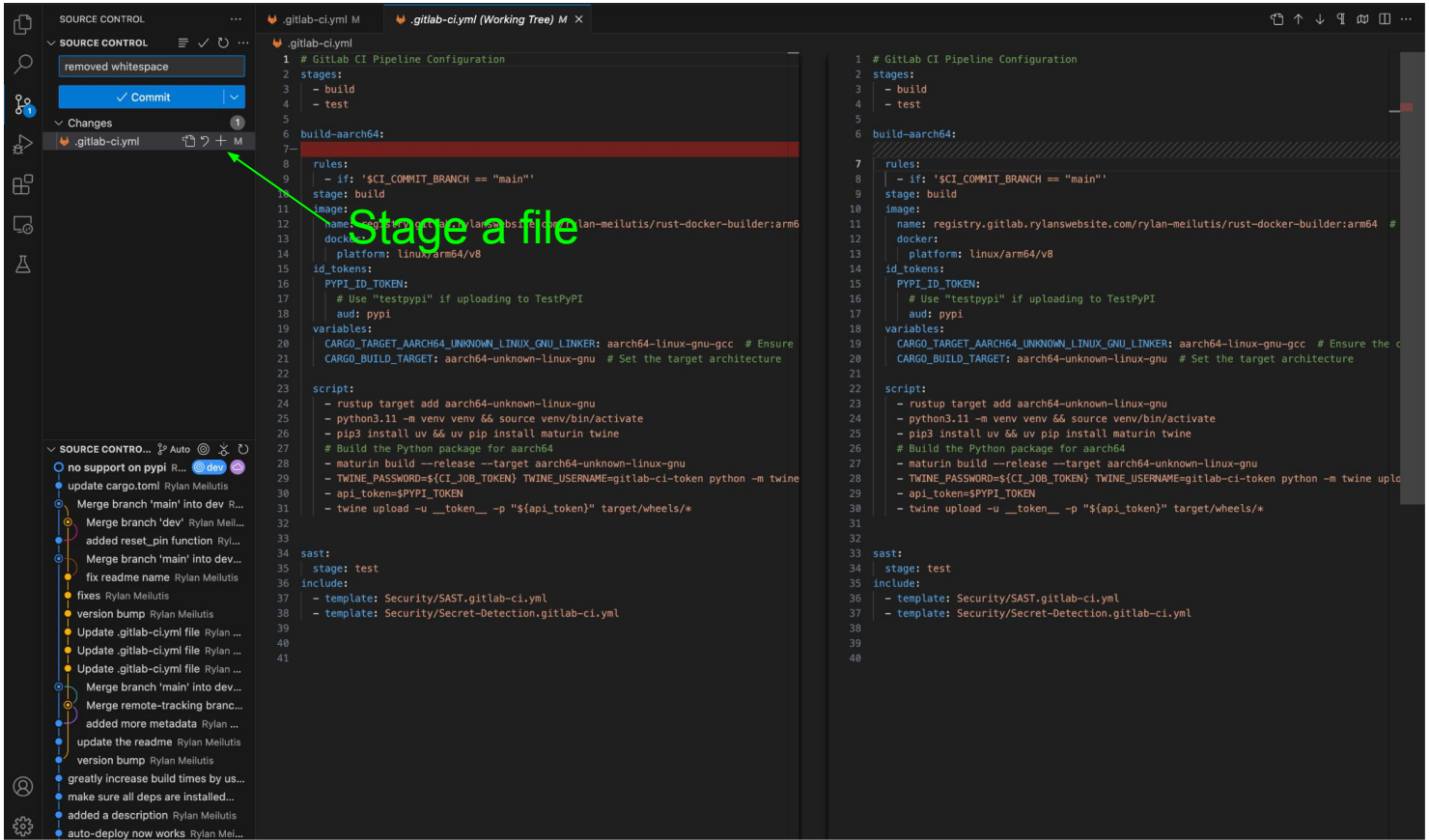
Lets you start a git repository locally on your machine to at a later time be pushed to a remote repository for more flexibility in access.

### git clone <remote-repo-url>

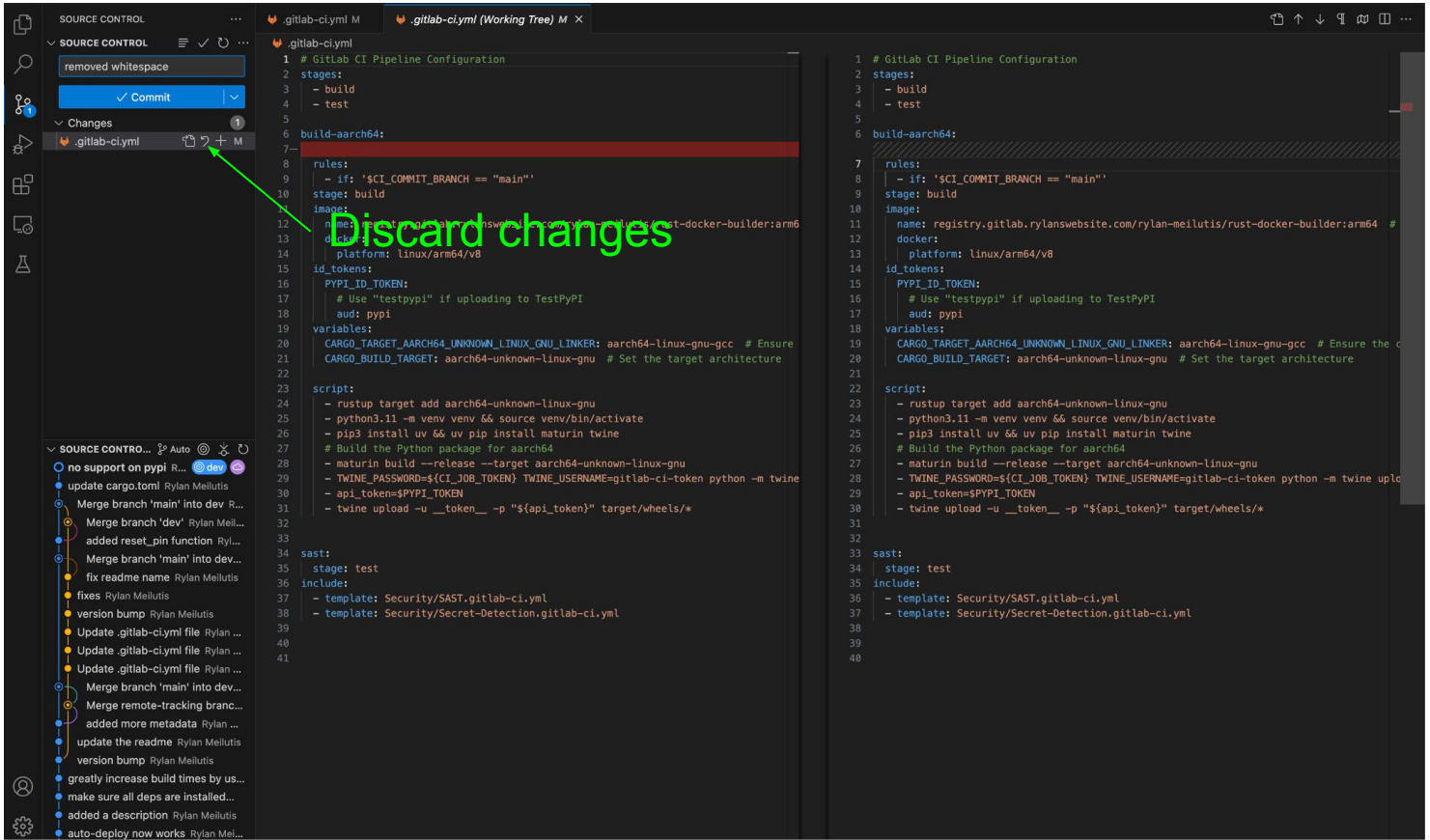
Clones an already made repository from a remote location

Lets you copy code over from a remote repo of your choice; ie. github, bitbucket, gitlabs.

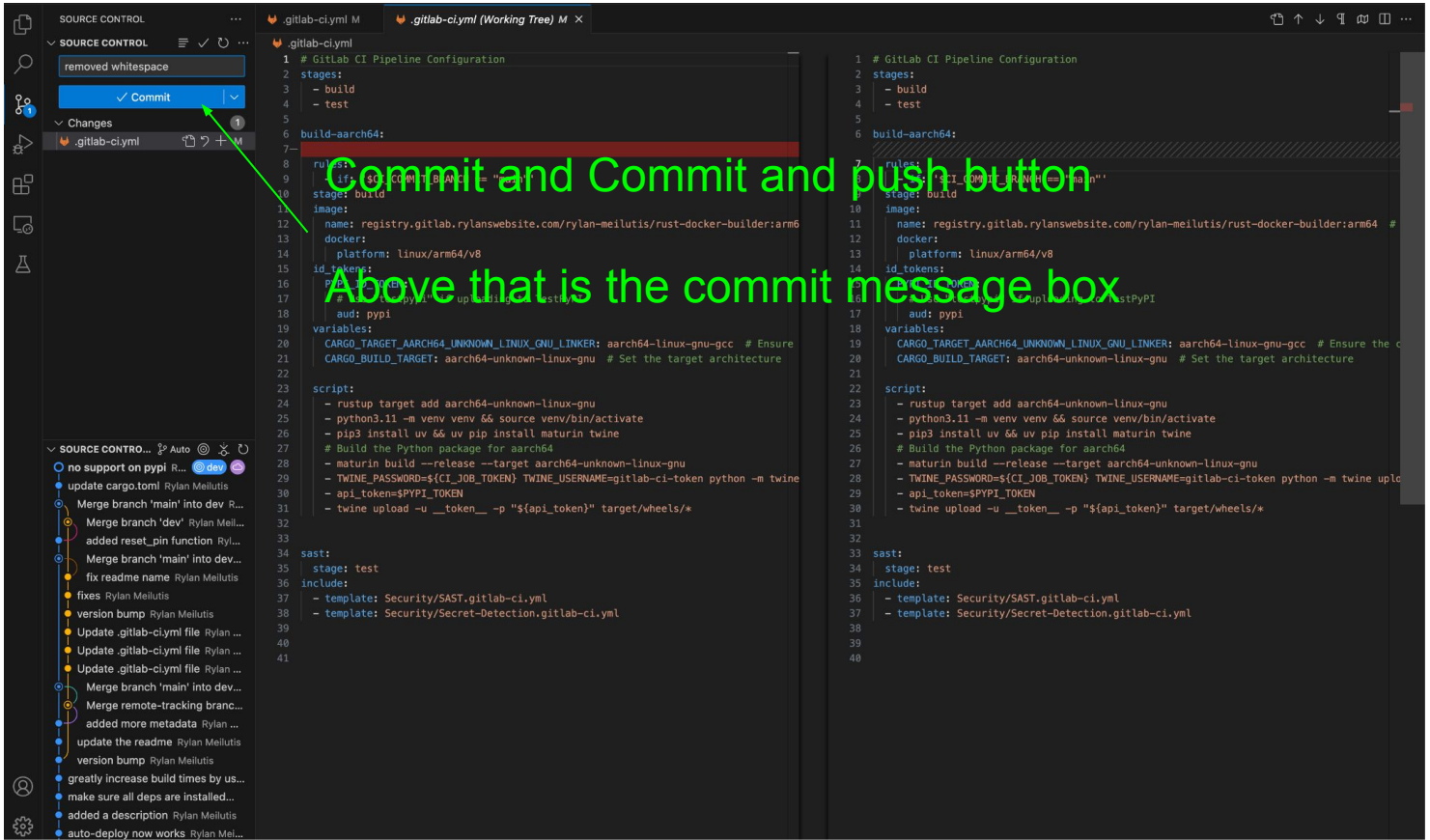
# IDE Integration: Vscode / Code Composer Studio



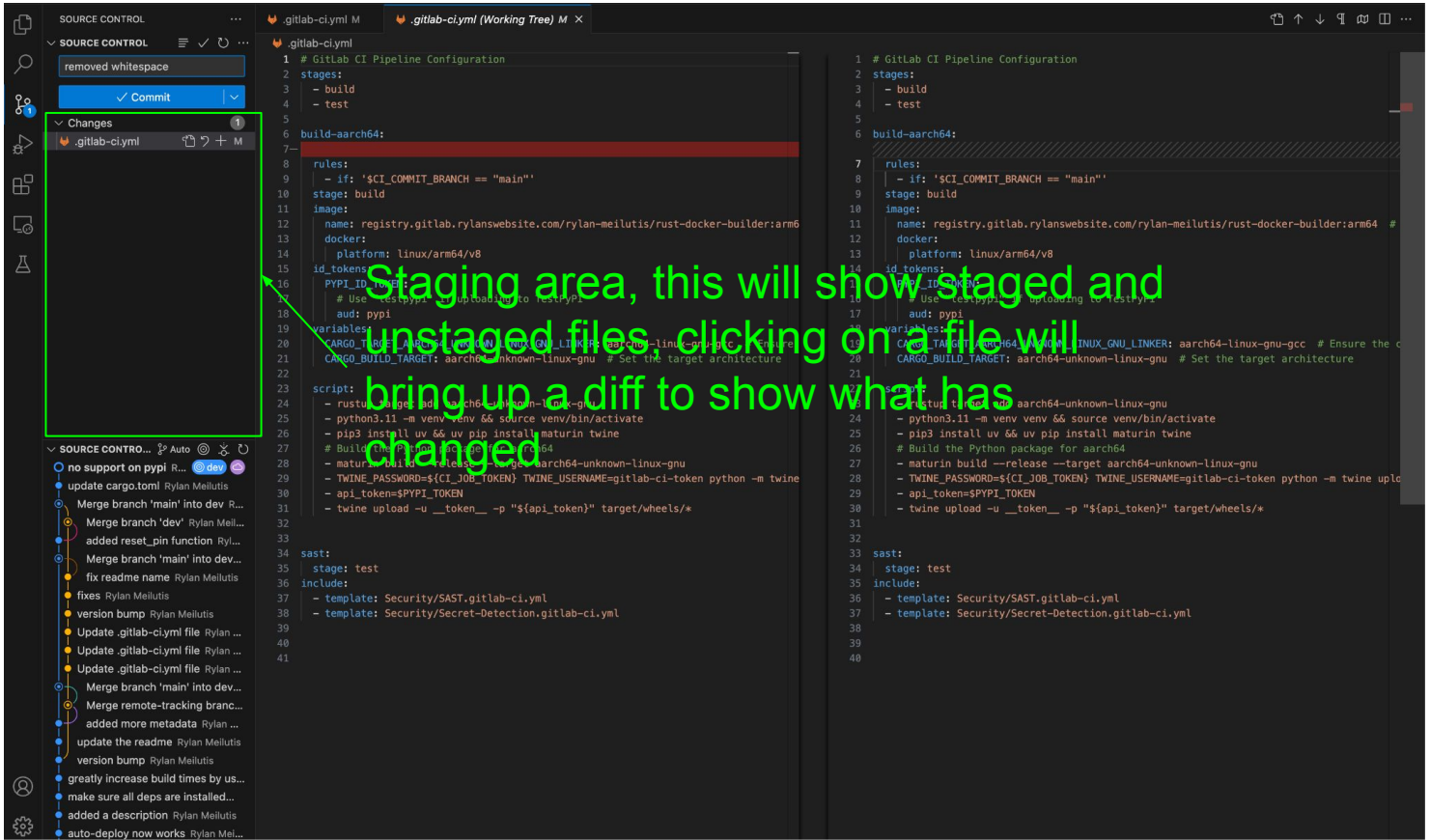
# IDE Integration: Vscode / Code Composer Studio



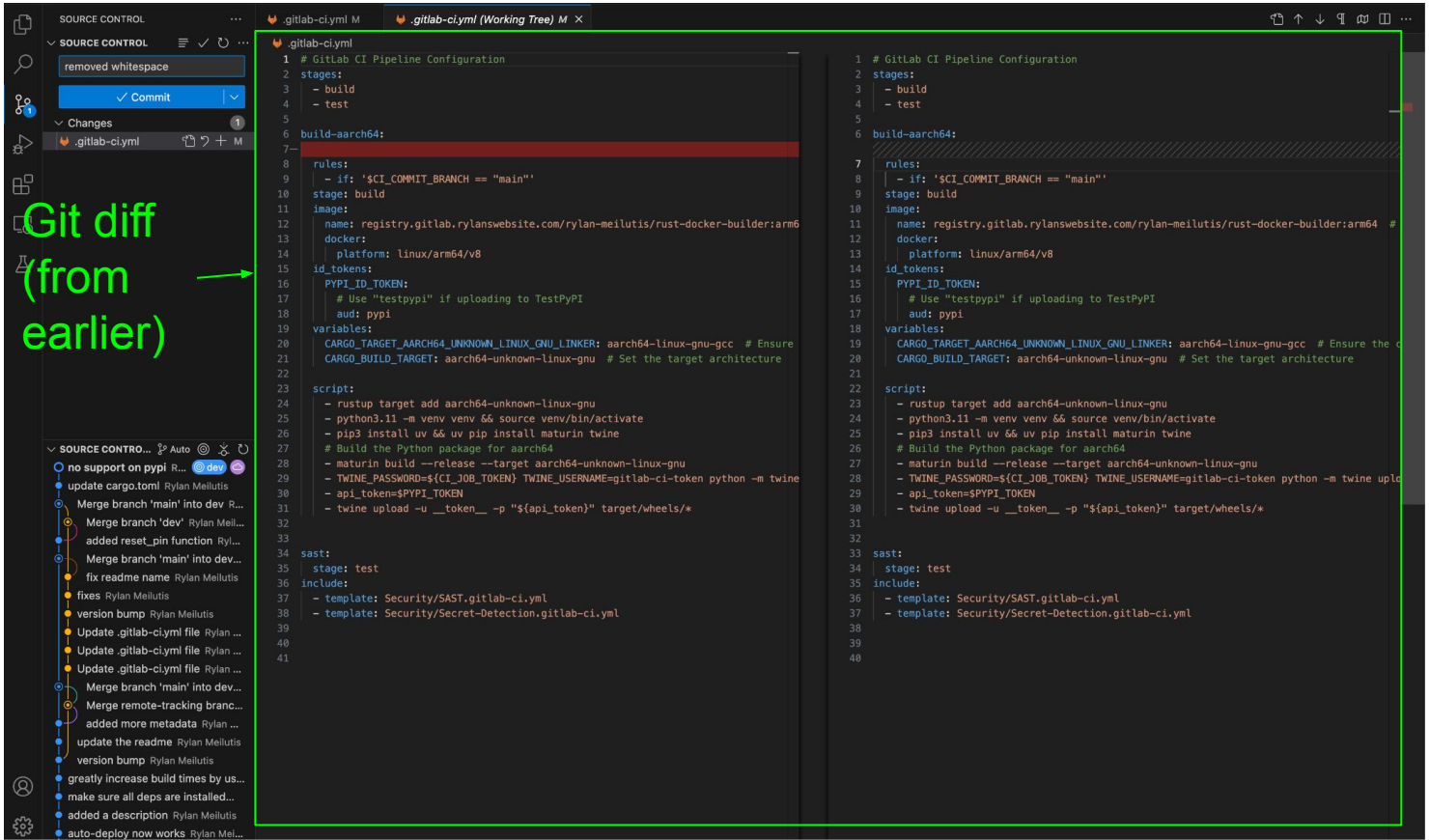
# IDE Integration: Vscode / Code Composer Studio



# IDE Integration: Vscode / Code Composer Studio



# IDE Integration: Vscode / Code Composer Studio



# IDE Integration: Vscode / Code Composer Studio

SOURCE CONTROL

removed whitespace

Commit

Changes 1

.gitlab-ci.yml M

.gitlab-ci.yml M | .gitlab-ci.yml (Working Tree) M

```yaml
# GitLab CI Pipeline Configuration
stages:
  - build
  - test

build-aarch64:

  rules:
    - if: '$CI_COMMIT_BRANCH == "main"'
  stage: build
  image:
    name: registry.gitlab.rylanswebsite.com/rylan-meilutis/rust-docker-builder:arm6
    docker:
      platform: linux/arm64/v8
  id_tokens:
    PYPI_ID_TOKEN:
      # Use "testpypi" if uploading to TestPyPI
      aud: pypi
  variables:
    CARGO_TARGET_AARCH64_UNKNOWN_LINUX_GNU_LINKER: aarch64-linux-gnu-gcc  # Ensure
    CARGO_BUILD_TARGET: aarch64-unknown-linux-gnu  # Set the target architecture

  script:
    - rustup target add aarch64-unknown-linux-gnu
    - python3.11 -m venv venv && source venv/bin/activate
    - pip3 install uv && uv pip install maturin twine
    # Build the Python package for aarch64
    - maturin build --release --target aarch64-unknown-linux-gnu
    - TWINE_PASSWORD=${CI_JOB_TOKEN} TWINE_USERNAME=gitlab-ci-token python -m twine
    - api_token=$PYPI_TOKEN
    - twine upload -u __token__ -p "${api_token}" target/wheels/*


sast:
  stage: test
include:
  - template: Security/SAST.gitlab-ci.yml
  - template: Security/Secret-Detection.gitlab-ci.yml
```

```yaml
# GitLab CI Pipeline Configuration
stages:
  - build
  - test

build-aarch64:

  rules:
    - if: '$CI_COMMIT_BRANCH == "main"'
  stage: build
  image:
    name: registry.gitlab.rylanswebsite.com/rylan-meilutis/rust-docker-builder:arm64  #
    docker:
      platform: linux/arm64/v8
  id_tokens:
    PYPI_ID_TOKEN:
      # Use "testpypi" if uploading to TestPyPI
      aud: pypi
  variables:
    CARGO_TARGET_AARCH64_UNKNOWN_LINUX_GNU_LINKER: aarch64-linux-gnu-gcc  # Ensure the c
    CARGO_BUILD_TARGET: aarch64-unknown-linux-gnu  # Set the target architecture

  script:
    - rustup target add aarch64-unknown-linux-gnu
    - python3.11 -m venv venv && source venv/bin/activate
    - pip3 install uv && uv pip install maturin twine
    # Build the Python package for aarch64
    - maturin build --release --target aarch64-unknown-linux-gnu
    - TWINE_PASSWORD=${CI_JOB_TOKEN} TWINE_USERNAME=gitlab-ci-token python -m twine uplo
    - api_token=$PYPI_TOKEN
    - twine upload -u __token__ -p "${api_token}" target/wheels/*


sast:
  stage: test
include:
  - template: Security/SAST.gitlab-ci.yml
  - template: Security/Secret-Detection.gitlab-ci.yml
```

SOURCE CONTRO... Auto

- no support on pypi R... dev
- update cargo.toml Rylan Meilutis
- Merge branch 'main' into dev R...
- Merge branch 'dev' Rylan Meil...
- added reset_pin function Ryl...
- Merge branch 'main' into dev...
- fix readme name Rylan Meilutis
- fixes Rylan Meilutis
- version bump Rylan Meilutis
- Update .gitlab-ci.yml file Rylan ...
- Update .gitlab-ci.yml file Rylan ...
- Update .gitlab-ci.yml file Rylan ...
- Merge branch 'main' into dev...
- Merge remote-tracking branc...
- added more metadata Rylan ...
- update the readme Rylan Meilutis
- version bump Rylan Meilutis
- greatly increase build times by us...
- make sure all deps are installed...
- added a description Rylan Meilutis
- auto-deploy now works Rylan Mei...

Source tree, this shows commits, merges, branches, tags and more.

# End of Presentation

Thank you for your time.

# Git - Looking Under the Hood: Intermediate Commands

Merging

Branching

Stashing

# Git - Looking Under the Hood: Common Issues

Divergent Branches

Local Branch Behind Main Branch

Merge Conflicts

Rebasing Conflicts

# Git - Looking Under the Hood: Common Issues

## Local Branch Behind Main Branch

git pull origin

Observe what files are producing conflict

Correct conflict by matching files in workspace with files in remote branch

git add <file-name>

Staging files without conflict puts you one step closer to resolving conflict

git push origin

Overcome conflict

# Git - Looking Under the Hood: Common Issues

Divergent Branches

git pull

- Encounter error

git fetch

git rebase

- No more error

git pull

git status

- Confirms everything up to date