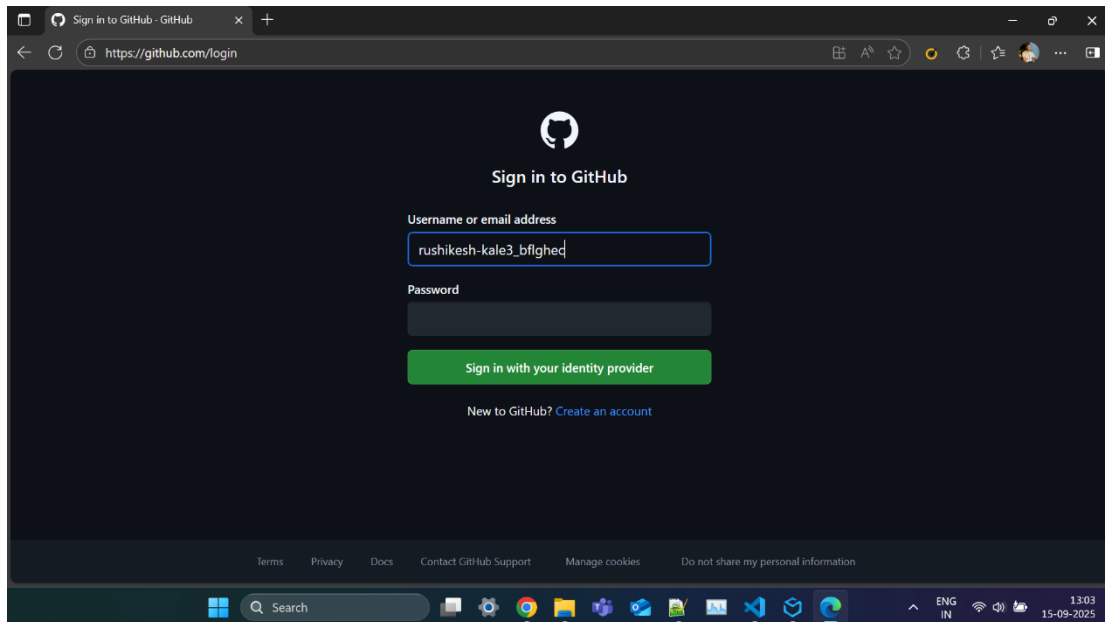


GitHub PR, Development And Deployment :

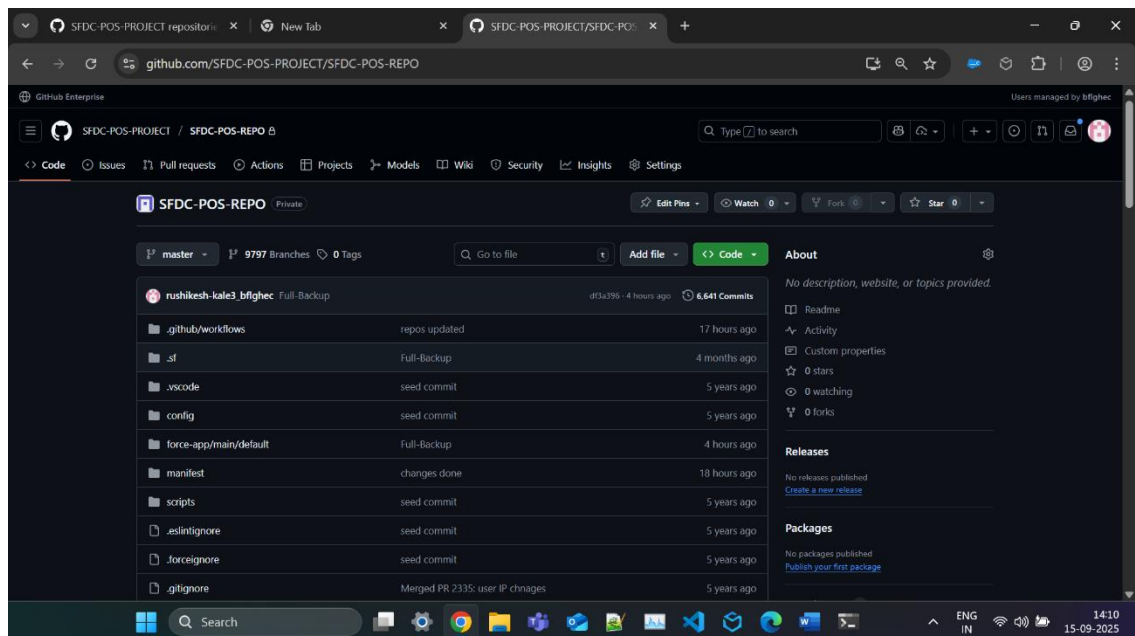
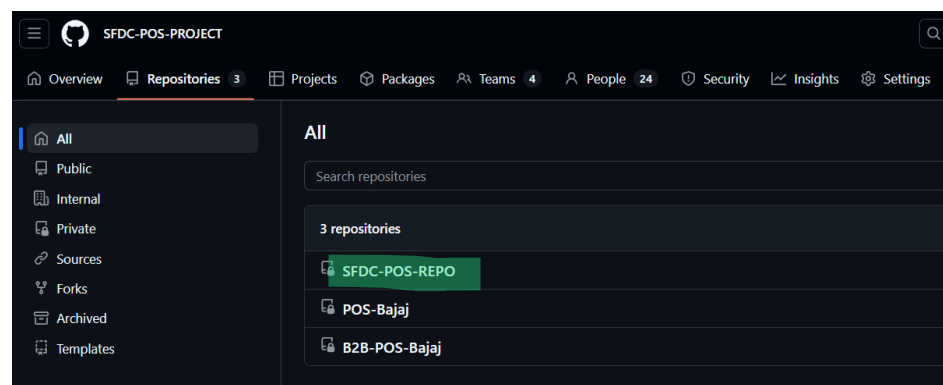
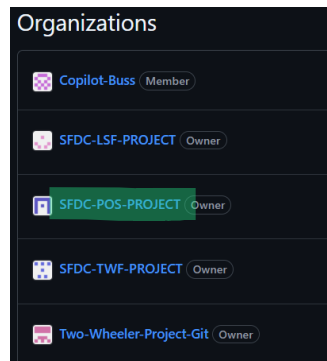
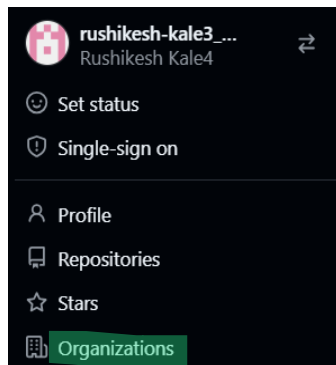
1. Login in the GitHub:

- <https://github.com> => GitHub_User_Name => Login with SOSL



2. Navigate to Repos :

- Profile => Organization -> Select Project => Select Repos



3. Create Personal Access Token :

- Profile => Setting => [Developer Setting](#) => Personal Access Token
- Generate New Classic Personal Access token => auth a git repo

The image shows the GitHub Developer Settings interface. On the left is a sidebar with navigation options: Repositories, Codespaces, Models (with a 'Preview' button), Packages, Copilot, Pages, Saved replies, Security (with 'Code security'), Integrations (with 'Applications' and 'Scheduled reminders'), and Archives (with 'Security log' and 'Sponsorship log'). At the bottom of the sidebar is a green 'Developer settings' button.

The main content area is titled 'Personal access tokens (classic)' and includes a 'Generate new token' button. It shows a list of tokens, with one selected: 'GH_PAT' with scopes 'admin:org, admin:ssh_signing_key, repo, workflow, write:packages' and a warning 'This token has no expiration date'. Below this is a 'Select scopes' section with a warning: 'GitHub strongly recommends that you set an expiration date for your token to help keep your information secure. Learn more'. The scopes are listed in a table:

Scope	Description
<input checked="" type="checkbox"/> repo	Full control of private repositories
<input checked="" type="checkbox"/> repo:status	Access commit status
<input checked="" type="checkbox"/> repo_deployment	Access deployment status
<input checked="" type="checkbox"/> public_repo	Access public repositories
<input checked="" type="checkbox"/> repo:invite	Access repository invitations
<input checked="" type="checkbox"/> security_events	Read and write security events
<input checked="" type="checkbox"/> workflow	Update GitHub Action workflows
<input checked="" type="checkbox"/> write:packages	Upload packages to GitHub Package Registry
<input checked="" type="checkbox"/> read:packages	Download packages from GitHub Package Registry
<input type="checkbox"/> delete:packages	Delete packages from GitHub Package Registry
<input checked="" type="checkbox"/> admin:org	Full control of orgs and teams, read and write org projects
<input checked="" type="checkbox"/> write:org	Read and write org and team membership, read and write org projects
<input checked="" type="checkbox"/> read:org	Read org and team membership, read org projects
<input checked="" type="checkbox"/> manage_runners:org	Manage org runners and runner groups

- Authorize the Repos for Token :

This screenshot shows the 'Personal access tokens (classic)' page with an authorization modal open. The page header includes 'Generate new token'. The main content shows a token 'GH_PAT' with scopes 'admin:org, admin:ssh_signing_key, repo, workflow, write:packages' and a warning 'This token has no expiration date'. The modal is titled 'Single sign-on organizations' and contains a search bar. Below the search bar, it lists organizations available for authorization:

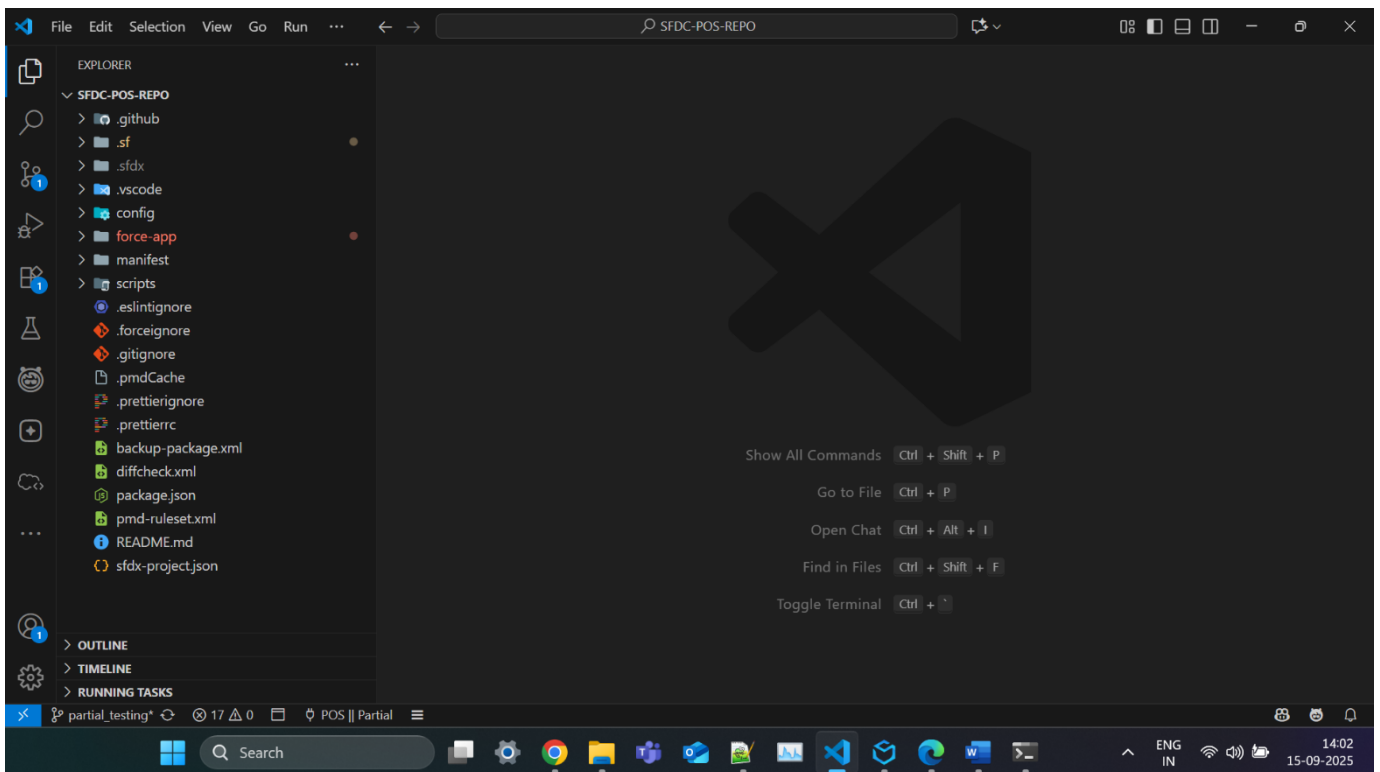
Organization	Action
Two-Wheeler-Project-Git	Authorize
Copilot-Buss	Authorize

4. Cloning Repos :

- Open terminal => `git clone GITHUB_URL`

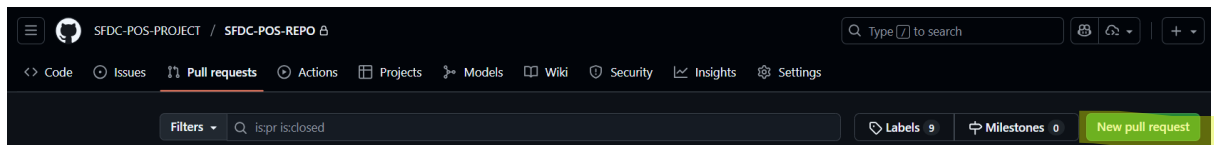
```
PS C:\Users\rushikeshkale3\Desktop\New folder (2)> git clone https://github.com/SFDC-POS-PROJECT/SFDC-POS-REPO.git
```

- Login in Terminal :
 - o window =>
 - Auto Logging => Browser Based Login
 - o mac =>
 - username : GitHub_User_Name
 - Pass : Personal Access Token generated in the Previous step
- Open Vs Code in the Cloned Folder :

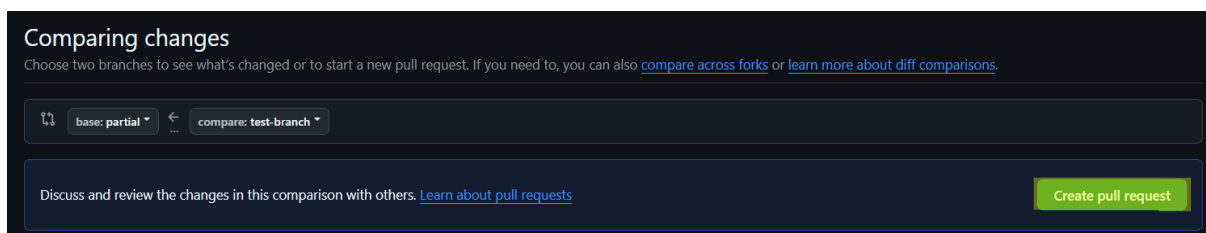


5. Raised the PR And Error Checking :

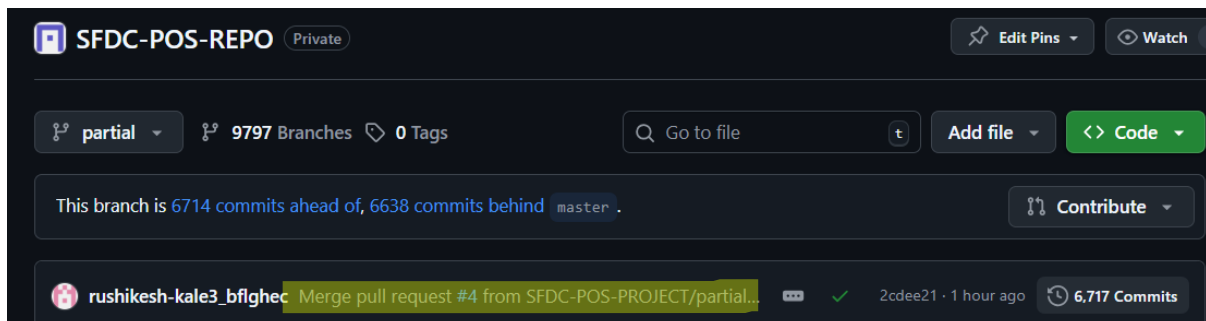
- Create Branch From Base Branch in VS Code[dev/partial/preprod/master]
- In your New Branch => Do Changes in the respective Classes and Code
- Update Package.xml file
- Commit the changes and Publish Your Branch
- Go to GitHub in PR Section



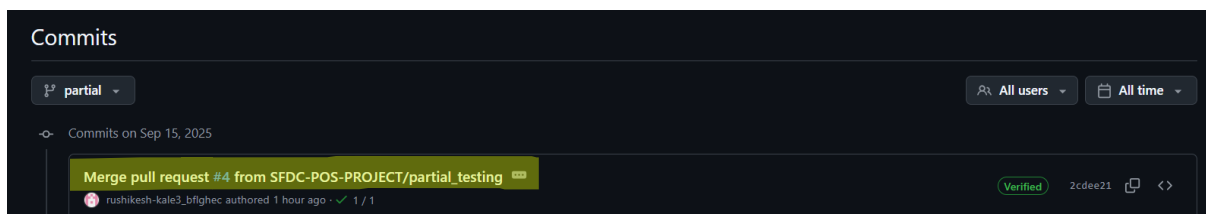
- Select you branch and base branch [like : test-branch into => partial]
- And Create Pull Request



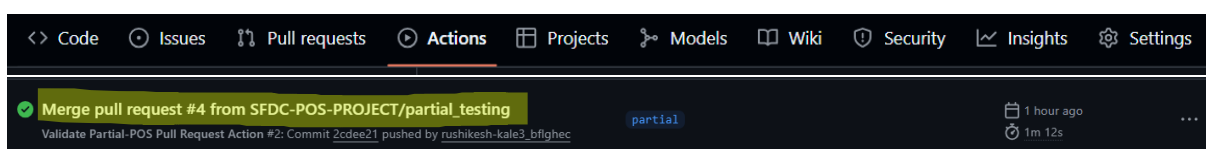
- After approval of the reviewer, You can Merge PR .
- Checking PR Commit and Release
- Code => select Base Branch => commits => your PR



- Check Merge Commit

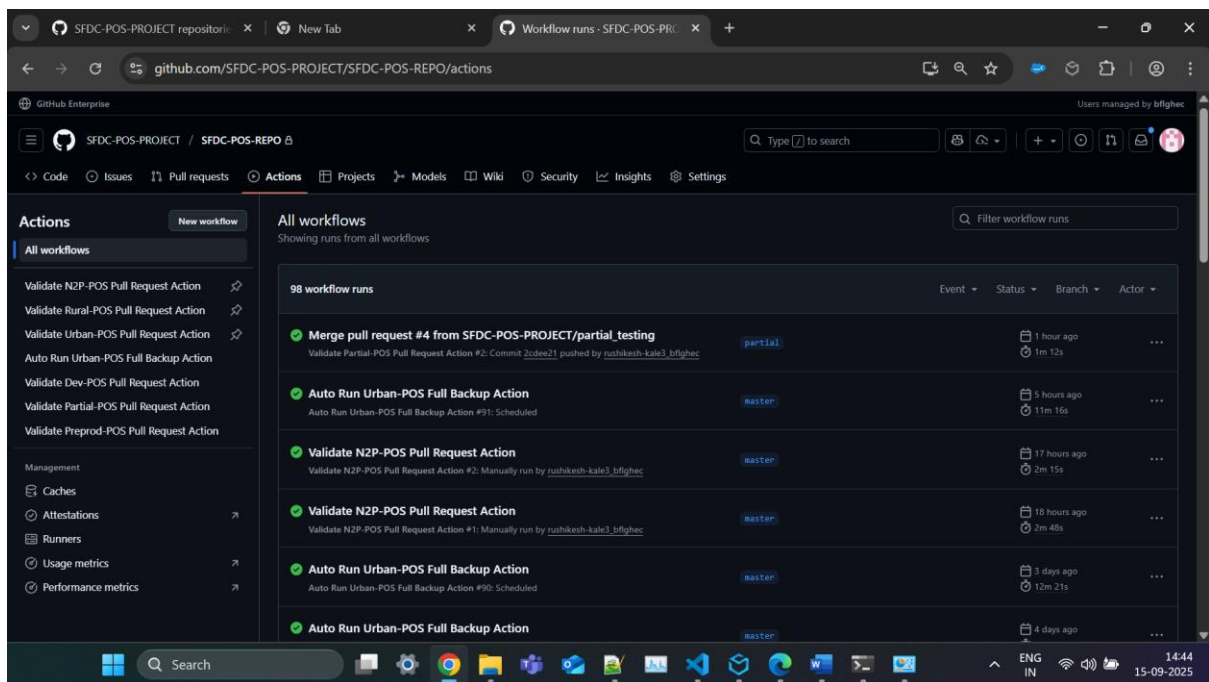


- Check Status of the Merging in the respective org :



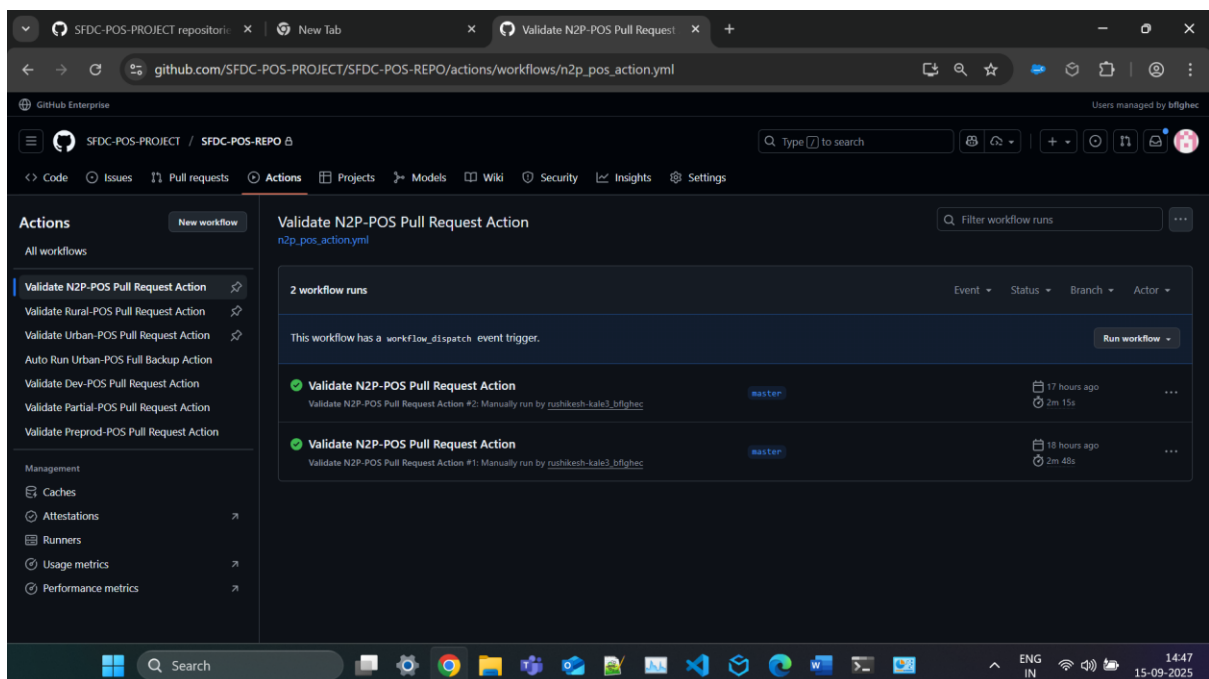
6. Production Deployment Details :

- Action Tab



- Select the Work Flow which you have to Run :

- Validate N2P-POS Pull Request Action – Rural N2P Package Validation
- Validate Rural-POS Pull Request Action – Rural Package Validation
- Validate Urban-POS Pull Request Action – Urban Package Validation
- Auto Run Urban-POS Full Backup Action – Urban Production Full Backup



- Run Workflow [If test String Required Put it space separated]:

The screenshot shows the GitHub Actions interface for the workflow 'Validate N2P-POS Pull Request Action' (n2p_pos_action.yml). It displays two workflow runs, both with a status of 'Success'. A modal is open, showing options to 'Run workflow' from the 'master' branch, with 'Test classes to run' set to 'caseTriggerOperationTest'.

Validate N2P-POS Pull Request Action
n2p_pos_action.yml

2 workflow runs

This workflow has a workflow_dispatch event trigger.

Run workflow

Use workflow from
Branch: master

Test classes to run *
caseTriggerOperationTest

Run workflow

- After Workflow Run, we can check the status by clicking on the particular action

The screenshot shows the details of the 'validate' job from the workflow 'Merge pull request #4 from SFDC-POS-PROJECT/partial_testing #2'. The job status is 'Success' and it took 1m 8s to complete. The job steps are listed, with 'Deploy to Environment' being the current step.

Validate Partial-POS Pull Request Action
Merge pull request #4 from SFDC-POS-PROJECT/partial_testing #2

Re-run all jobs

Summary

Jobs

validate

Run details

Usage

Workflow file

validate
succeeded 2 hours ago in 1m 8s

Search logs

Set up job 2s

Run actions/checkout@v3 7s

Install Salesforce CLI 16s

Login to Environment 5s

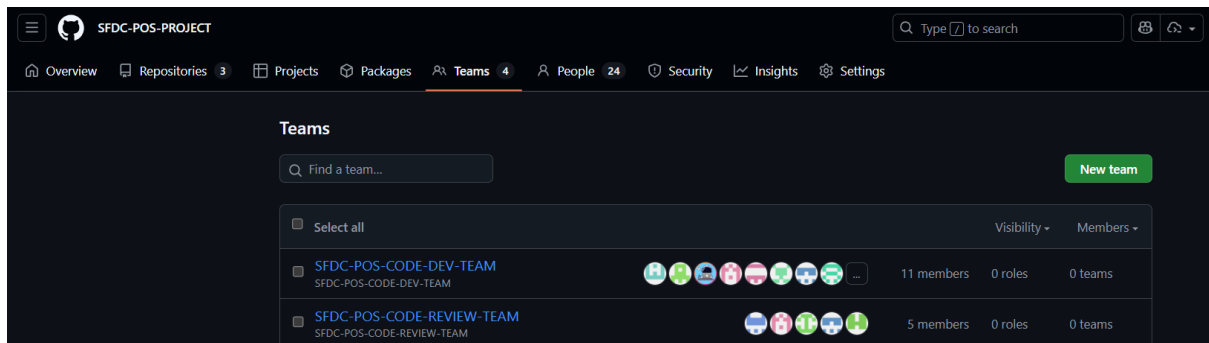
Deploy to Environment 37s

Post Run actions/checkout@v3 8s

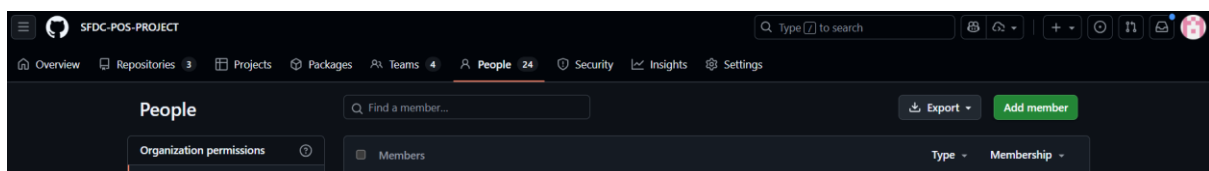
Complete job 8s

7. Admin Part :

- Add user According to Roles :
 - o Owner -> People -> Add Member -> Owner
 - o Code Review -> Teams -> [SFDC-POS-CODE-REVIEW-TEAM](#)
 - o Developer -> Teams -> [SFDC-POS-CODE-DEV-TEAM](#)
- Developer and Code Review :



- Owner :



- Permission For Roles :
 - o Owner -> Can Add New Member
 - o Code Review -> Can Approve PR
 - o Developer -> Raised the PR

8. Note :

1. Git Command :
 - a. `git clone {GITHUB_URL}` => cloning into local machine
 - b. `git fetch --all` => update old local repos
 - c. `git remote -v` => check added remote URL
 - d. `git remote set-url origin NEW_URL_WITHTOKEN`
2. It Is Good Practice to Set Remote with Personal Access Token :
 - a. `git remote -v` => will get URL

```
PS C:\Users\rushikeshkale3\Desktop\New folder (2)\SFDC-POS-REPO> git remote -v
origin https://github.com/SFDC-POS-PROJECT/SFDC-POS-REPO.git (fetch)
origin https://github.com/SFDC-POS-PROJECT/SFDC-POS-REPO.git (push)
```

- b. `NEW_URL_WITHTOKEN` = `https://` 'access_token@' existing_url[from github]
 - c. `git remote set-url origin NEW_URL_WITHTOKEN`
3. Error : [repository not found] :
 - a. Navigate to : Control Panel => User Accounts => Credential Manager => Remove For GitHub

