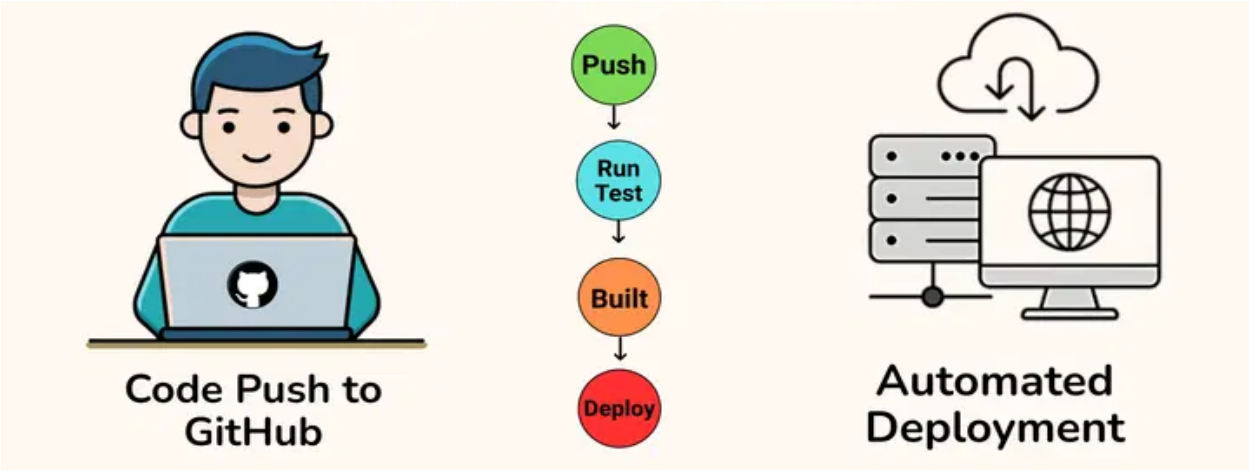
End-to-End CI/CD Workflow with GitHub Runner & Docker Compose



By
Rahul Ahmed

Introduction

This document provides a complete guide for setting up a Continuous Integration and Continuous Deployment (CI/CD) pipeline using GitHub Actions with a self-hosted runner.

The pipeline automates the entire process — from building and pushing Docker images to deploying updated services on the server using Docker Compose.

Specifically designed for the one service (topup-management-service), this implementation ensures faster, reliable, and version-controlled deployments.
It also supports backup, rollback, and health checks to maintain high availability and minimal downtime during releases.

This documentation serves as a reference for DevOps engineers and developers who want to replicate or extend the same setup for other docker services..

Step 1

Setup and configure Github Self-hosted runner

Create a directory for runner:

mkdir -p /home/rahul/actions-runner
cd /home/rahul/actions-runner

Download and extract runner file

curl -L -o actions-runner-linux-x64-2.328.0.tar.gz https://github.com/actions/runner/releases/download/v2.328.0/actions-runner-linux-x64-2.328.0.tar.gz

tar xzf actions-runner-linux-x64-2.328.0.tar.gz

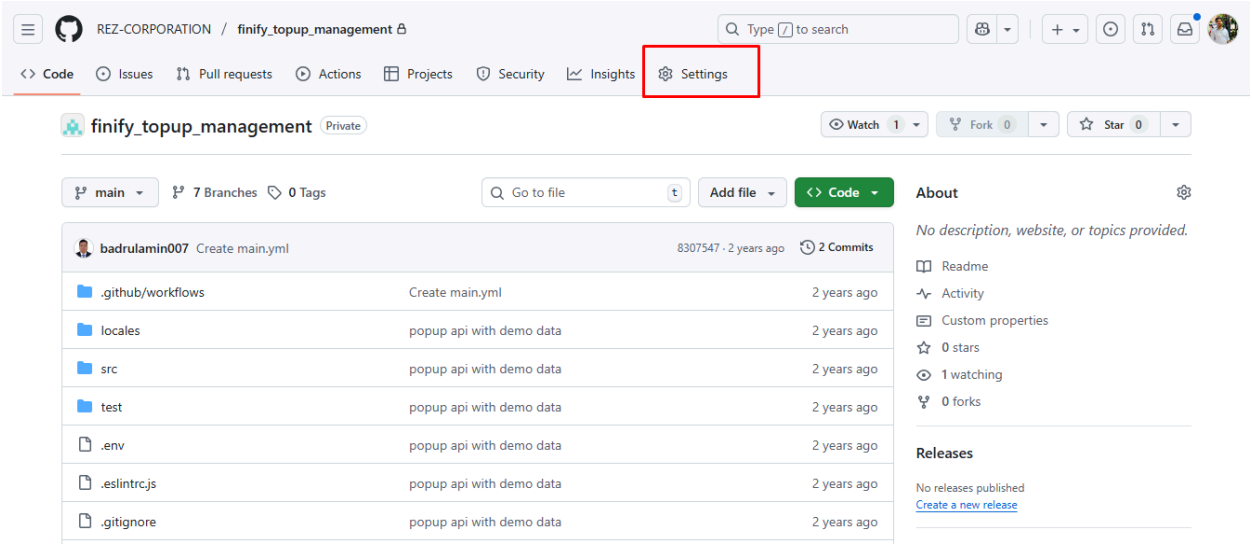
Step 2

Create an server user named ***githubrunner*** we will this user account for continues deployment.

Step 3

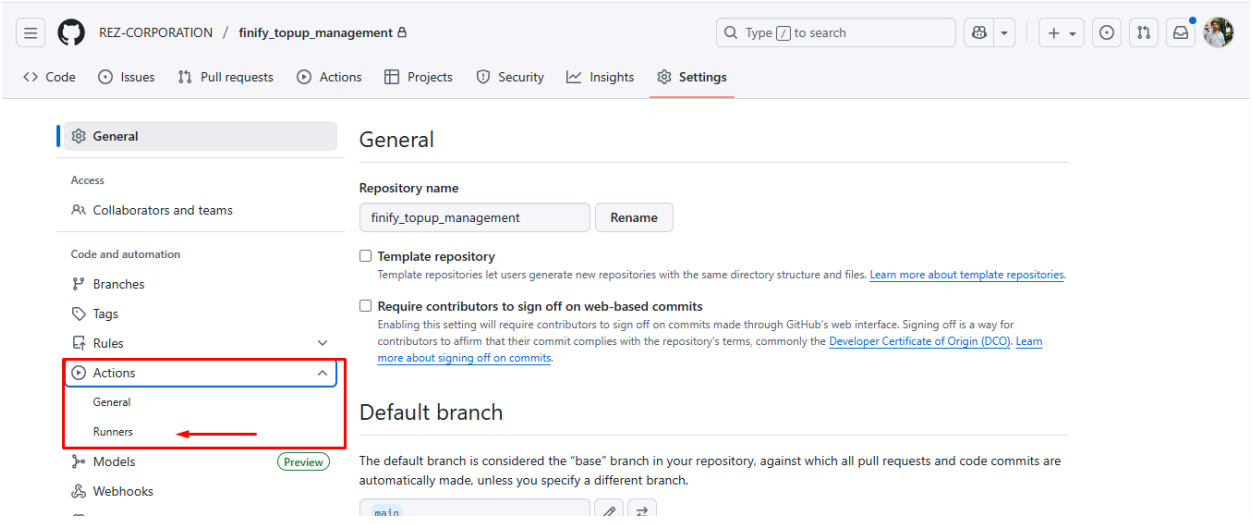
Configure repo for runner

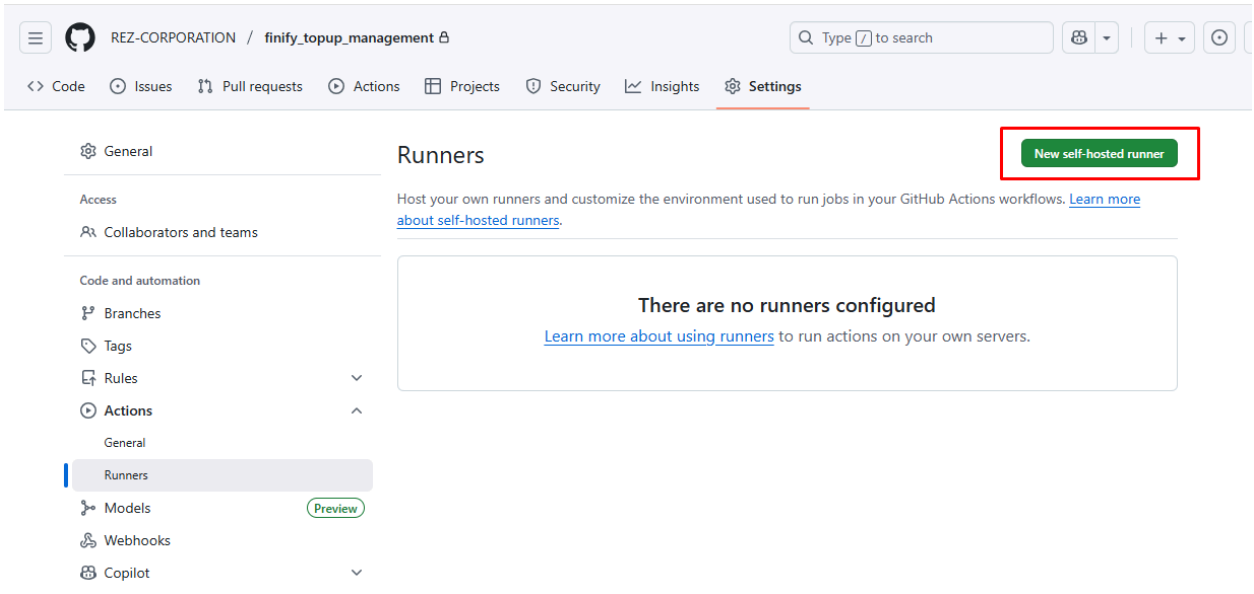
Goto github repo setting



Step 4

Go to **Settings>Action>Runner** under setting and click on New **self-hosted runner** button showed in screenshot





Step 5

Click on Linux as we use linux server. Then follow the configure part. In configure part repo token is shared.

Configure

```
# Create the runner and start the configuration experience
$ ./config.sh --url https://github.com/REZ-CORPORATION/finify_topup_management --token
BGJLLZOE7H423EHL3L7W5ODJBBYYW

# Last step, run it!
$ ./run.sh
```

Step 6

Copy the command and run with ***githubrunner*** user account. After run the command it will ask few questions you can skip those.

```
githubrunner@ubuntu-32gb-hel1-3:/home/rahul/actions-runner$ ./config.sh --url https://github.com/REZ-CORPORATION/finify_topup_management --token BGJLLZOE7H423EHL3L7W5C
DJBBYYW

--------------------------------------------------------------------------------
|        ____ _ _   _   _       _          _     _                            |
|       / ___(_) |_| | | |_   _| |__      / \   ___| |_(_) ___  _ __  ___      |
|      | |  _| | __| |_| | | | | '_ \    / _ \ / __| __| |/ _ \| '_ \/ __|     |
|      | |_| | | |_|  _  | |_| | |_) |  / ___ \ (__| |_| | (_) | | | \__ \     |
|       \____|_|\__|_| |_|\__,_|_.__/  /_/   \_\___|\__|_|\___/|_| |_|___/     |
|                                                                              |
|                       Self-hosted runner registration                        |
|                                                                              |
--------------------------------------------------------------------------------

# Authentication


√ Connected to GitHub

# Runner Registration

Enter the name of the runner group to add this runner to: [press Enter for Default]

Enter the name of runner: [press Enter for ubuntu-32gb-hel1-3] Dev-server-227

This runner will have the following labels: 'self-hosted', 'Linux', 'X64'
Enter any additional labels (ex. label-1,label-2): [press Enter to skip]

√ Runner successfully added
√ Runner connection is good

# Runner settings

Enter name of work folder: [press Enter for _work] finify-deployement

√ Settings Saved.
```

Step 7

Run the command *./run.sh* and output will show like this:

```
githubrunner@ubuntu-32gb-hel1-3:/home/rahul/actions-runner$ ./run.sh

√ Connected to GitHub

Current runner version: '2.328.0'
2025-11-03 08:26:25Z: Listening for Jobs
Runner update in progress, do not shutdown runner.
Downloading 2.329.0 runner
Waiting for current job finish running.
Generate and execute update script.
Runner will exit shortly for update, should be back online within 10 seconds.
Runner update process finished.
Runner listener exit because of updating, re-launch runner after successful update
Update finished successfully.
Restarting runner...

√ Connected to GitHub

Current runner version: '2.329.0'
```

Step 8

If you want to run this as a service to make sure it auto-starts after reboot follow these commands:

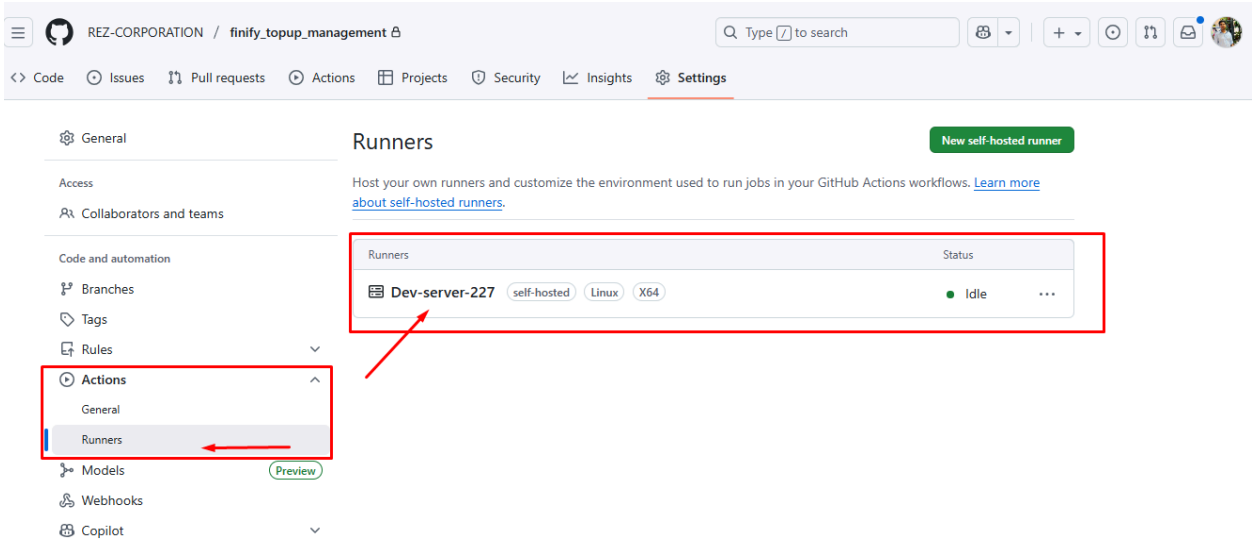
For config: *sudo ./svc.sh install*

Start service *sudo ./svc.sh start*

Check status *sudo ./svc.sh status*

Step 9

Now go to Runners under Action in github repo setting, you will found your server is showing



Step 10

Need to update existing github workflow file for Continues Deployment. As we already use only Continous integration so we need to update the Continues Deployment part only. Here is the workflow file:

Only CD & Health Check Part

```
deploy:
  runs-on: self-hosted
  needs: build
  steps:
   - name: Print deploy info
     run: |
      echo "Deploying service: $SERVICE_NAME"
```

```
        echo "Compose path: $COMPOSE_PATH"
        echo "Image: $IMAGE_NAME:${{ github.run_number }}"

    - name: Pull latest image
      run: docker pull $IMAGE_NAME:${{ github.run_number }}

    - name: Backup current docker-compose.yml
      run: cp $COMPOSE_PATH ${COMPOSE_PATH}.bak

    - name: Update image tag in compose file
      run: |
        sed -i "s#image: $IMAGE_NAME:.*#image: $IMAGE_NAME:${{ github.run_number }}#g" $COMPOSE_PATH
        echo "Updated image tag to: $IMAGE_NAME:${{ github.run_number }}"

    - name: Deploy service
      working-directory: /home/ubuntu/finify/INTEGRATIONS/topup
      run: |
        echo "Deploying $SERVICE_NAME..."
        docker compose up -d --no-deps --build $SERVICE_NAME
        echo "Service restarted."

    - name: Health check
      run: |
        sleep 10
          HEALTH=$(docker inspect --format='{{.State.Health.Status}}' $(docker ps -q --filter name=$SERVICE_NAME)
2>/dev/null || echo "none")
        echo "Health status: $HEALTH"
        if [ "$HEALTH" = "unhealthy" ]; then
          echo "⚠️ Health check failed. Rolling back..."
          mv ${COMPOSE_PATH}.bak $COMPOSE_PATH
          docker compose up -d --no-deps --build $SERVICE_NAME
          exit 1
        fi

    - name: Cleanup backup file
      if: always()
      run: rm -f ${COMPOSE_PATH}.bak || true
```

Remarks: Need to mention the deployment docker-compose.yml file, path and container name in workflow file.

Full workflow file:

```
name: Build & Deploy Finify Topup Management (Dev)

on:
  push:
    branches: [ finify-dev ]

env:
  SERVICE_NAME: finify-topup-management-dev
  COMPOSE_PATH: /home/ubuntu/finify/INTEGRATIONS/topup/docker-compose.yml
  IMAGE_NAME: rosebaysystems/finify-topup-management-dev

jobs:
  build:
    runs-on: ubuntu-latest

    steps:
```

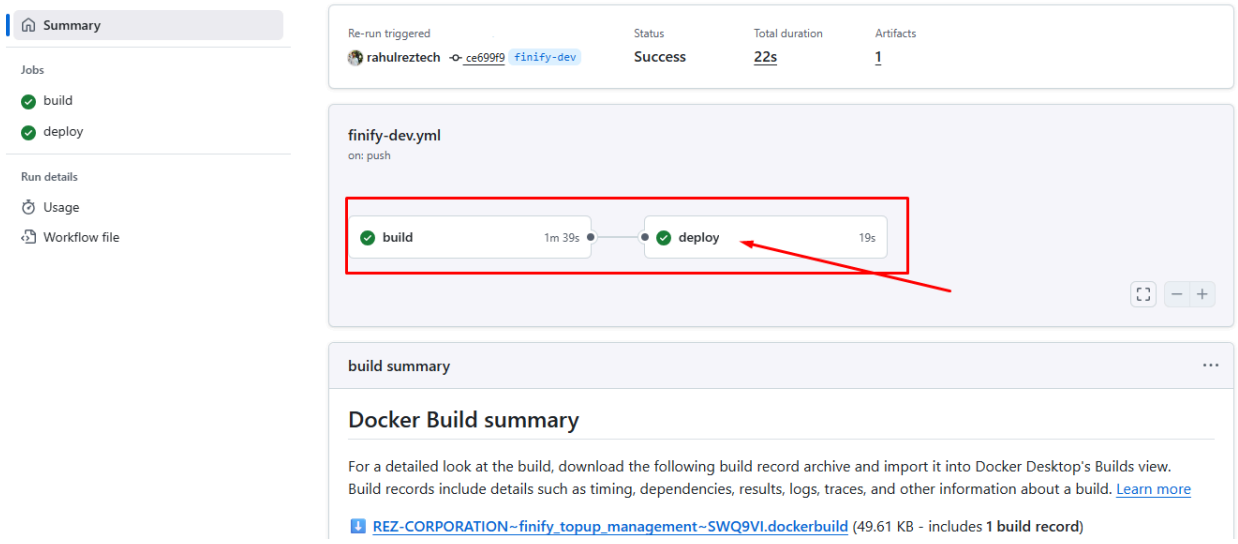
```
      - name: Checkout repository
        uses: actions/checkout@v4

      - name: Login to Docker Hub
        uses: docker/login-action@v3
        with:
          username: ${{ secrets.DOCKER_HUB_USERNAME }}
          password: ${{ secrets.DOCKER_HUB_PASSWORD }}

      - name: Set up Docker Buildx
        uses: docker/setup-buildx-action@v3

      - name: Build and Push Docker Image
        uses: docker/build-push-action@v6
        with:
          context: .
          file: ./Dockerfile
          push: true
          tags: ${{ env.IMAGE_NAME }}:${{ github.run_number }}

  deploy:
    runs-on: self-hosted
    needs: build
    steps:
      - name: Print deploy info
        run: |
          echo "Deploying service: $SERVICE_NAME"
          echo "Compose path: $COMPOSE_PATH"
          echo "Image: $IMAGE_NAME:${{ github.run_number }}"

      - name: Pull latest image
        run: docker pull $IMAGE_NAME:${{ github.run_number }}

      - name: Backup current docker-compose.yml
        run: cp $COMPOSE_PATH ${COMPOSE_PATH}.bak

      - name: Update image tag in compose file
        run: |
          sed -i "s#image: $IMAGE_NAME:.*#image: $IMAGE_NAME:${{ github.run_number }}#g" $COMPOSE_PATH
          echo "Updated image tag to: $IMAGE_NAME:${{ github.run_number }}"

      - name: Deploy service
        working-directory: /home/ubuntu/finify/INTEGRATIONS/topup
        run: |
          echo "Deploying $SERVICE_NAME..."
          docker compose up -d --no-deps --build $SERVICE_NAME
          echo "Service restarted."

      - name: Health check
        run: |
          sleep 10
          HEALTH=$(docker inspect --format='{{.State.Health.Status}}' $(docker ps -q --filter name=$SERVICE_NAME)
2>/dev/null || echo "none")
          echo "Health status: $HEALTH"
          if [ "$HEALTH" = "unhealthy" ]; then
            echo "⚠️ Health check failed. Rolling back..."
            mv ${COMPOSE_PATH}.bak $COMPOSE_PATH
            docker compose up -d --no-deps --build $SERVICE_NAME
            exit 1
          fi

      - name: Cleanup backup file
        if: always()
```

```
run: rm -f ${COMPOSE_PATH}.bak || true
```

Step 11

Boom

Now the deployment automation done. First it will build the service then deploy it automatically in server. If deployment failed at any stage you will notify by mail, also if container exited to any issue happened and caught on docker health check it will automatically rollback the tag immediately.



Thank You