

EXP-5.2:

AIM:

Understand how to automate testing and deployment using GitHub Actions. Learn to trigger workflows on code pushes, run tests automatically, and deploy artifacts to your hosting environment.

PROCEDURE-

Step 1: Create Workflow File

- In your repository, create a directory named `.github/workflows` if it does not already exist.
- Inside `.github/workflows`, create a file named `main.yml`.

Step 2: Add Workflow Configuration

- Copy and paste the following YAML configuration into `main.yml`:

```
```yaml
name: CI/CD Pipeline
on:
 push:
 branches:
 - main
jobs:
 build-test-deploy:
 runs-on: ubuntu-latest
 steps:
 - name: Checkout code
 uses: actions/checkout@v4
 - name: Set up Node.js
 uses: actions/setup-node@v4
 with:
 node-version: 18
 - name: Install dependencies
```

- run: npm install
- name: Run tests
- run: npm test
- name: Build project
- run: npm run build

- This configuration triggers the workflow whenever you push to the main branch, sets up Node.js, installs dependencies, runs tests, and builds the project.

### **Step 3: Commit and Push**

- Stage and commit the workflow file:

```
```sh
git add .github/workflows/main.yml
git commit -m "Add CI/CD pipeline workflow"
git push origin main
```
```

- This will upload the workflow file to your repository.

### **Step 4: Observe Workflow Runs**

- Go to your repository on GitHub.
- Click on the "Actions" tab.
- You'll see your workflow running after the push. Status will display as queued, running, succeeded, or failed.

### **Step 5: Configure Deployment (OPTIONAL)**

Deploy to GitHub Pages

- Uncomment and add these steps to your workflow if you want to deploy to GitHub Pages:

```
```yaml
- name: Deploy to GitHub Pages
  uses: peaceiris/actions-gh-pages@v4
  with:
```

```
github_token: ${{ secrets.GITHUB_TOKEN }}
publish_dir: ./build
...
```

- Make sure your build artifacts (static files) go into the `./build` directory.

Deploy to Netlify or AWS S3

- For Netlify, use the official Netlify Action and add required secrets (see Netlify docs).
- For AWS S3, use the official S3 Action and add AWS credentials to your repo secrets.

Step 6: Test the Automation

- Make a small code change and push it to the main branch.
- Monitor the workflow status in the "Actions" tab.
- Confirm that tests run and builds are produced automatically. If you set up deployment, verify that your site is updated.

Step 7: Create the Workflow Directory and File

- In your project's root directory, add a folder called `.github`, and inside that, another folder called `workflows`.
- In `.github/workflows`, create a new file named `main.yml`.

Step 8: Configure the Workflow File

- Paste this configuration into `.github/workflows/main.yml`:

```
```yaml
name: CI/CD Pipeline
on:
 push:
 branches:
 - main
jobs:
 build-test-deploy:
```

```
runs-on: ubuntu-latest
steps:
 - name: Checkout code
 uses: actions/checkout@v4
 - name: Set up Node.js
 uses: actions/setup-node@v4
 with:
 node-version: 18
 - name: Install dependencies
 run: npm install
 - name: Run tests
 run: npm test
 - name: Build project
 run: npm run build
```

```
Optional Deploy to GitHub Pages
- name: Deploy to GitHub Pages
uses: peaceiris/actions-gh-pages@v4
with:
github_token: ${ secrets.GITHUB_TOKEN }
publish_dir: ./build
```

...

- This will check out the code, set up Node.js, run `npm install`, `npm test`, and `npm run build` on each push to main.

## Step 9: Commit and Push Your Workflow

- Save all your changes, then run these commands:

```
``sh
git add .github/workflows/main.yml
git commit -m "Add GitHub Actions CI/CD pipeline"
git push origin main
...
```

- This updates your GitHub repository with the workflow file.

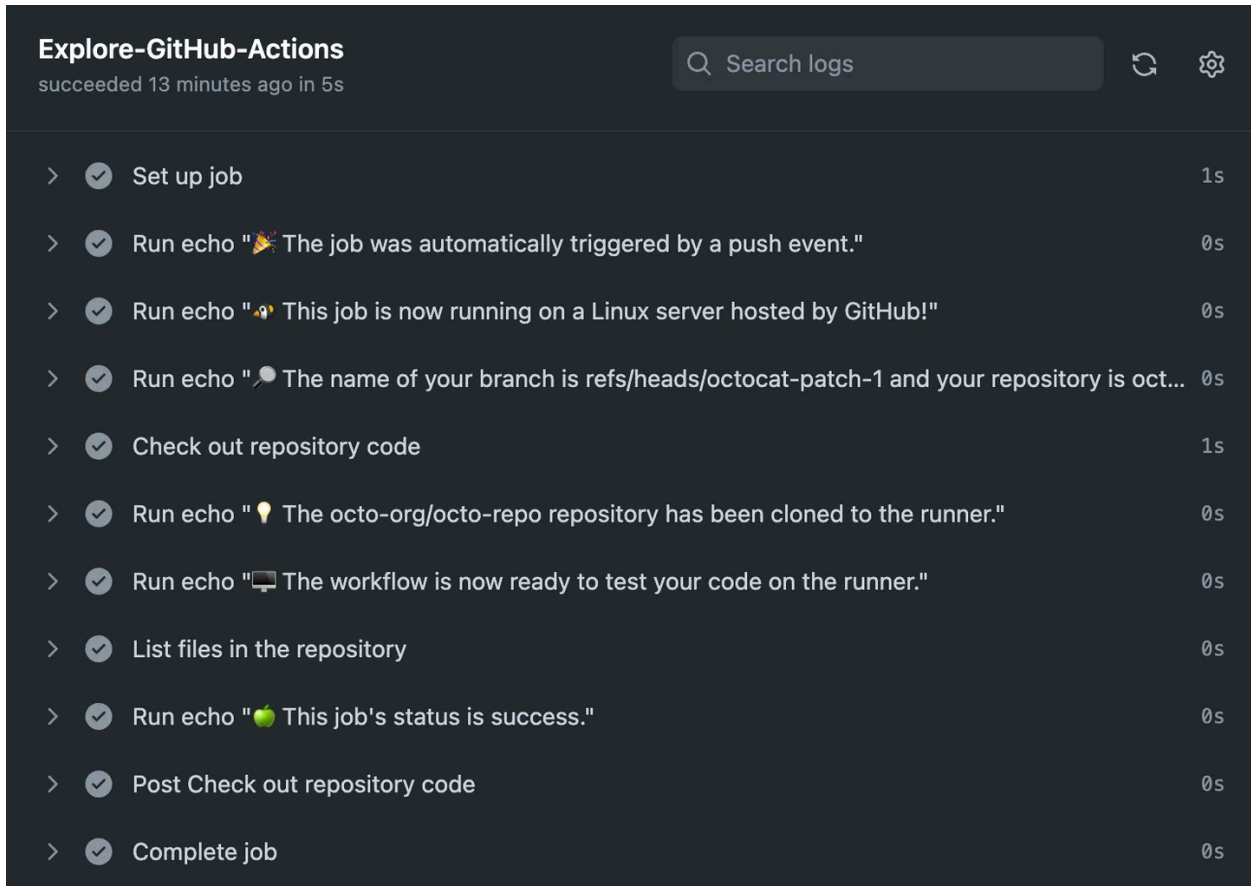
## Step 10: View Workflow Status in GitHub

- Go to your repository on GitHub.
- Click on the "Actions" tab next to "Code."
- You'll see your workflow listed, and a run will start automatically for your latest push.

## Step 11: Configure Deployment (Optional)

- If you want to deploy to GitHub Pages, make sure your build outputs to a `build` directory.
- In your workflow file, uncomment the deployment step and adjust `publish\_dir` if needed.
- For Netlify or S3, use their respective deployment actions and add secrets/tokens as needed.

## OUTPUT:



The screenshot displays the GitHub Actions interface for a workflow named "Explore-GitHub-Actions". The status is "succeeded 13 minutes ago in 5s". A search bar labeled "Search logs" is present. The workflow consists of 12 steps, all of which are completed successfully, indicated by green checkmarks. The steps are as follows:

Step	Duration
> ✓ Set up job	1s
> ✓ Run echo "🚀 The job was automatically triggered by a push event."	0s
> ✓ Run echo "👋 This job is now running on a Linux server hosted by GitHub!"	0s
> ✓ Run echo "💬 The name of your branch is refs/heads/octocat-patch-1 and your repository is oct..."	0s
> ✓ Check out repository code	1s
> ✓ Run echo "💡 The octo-org/octo-repo repository has been cloned to the runner."	0s
> ✓ Run echo "💻 The workflow is now ready to test your code on the runner."	0s
> ✓ List files in the repository	0s
> ✓ Run echo "🍏 This job's status is success."	0s
> ✓ Post Check out repository code	0s
> ✓ Complete job	0s