GitHub Actions Setup Guide for Automated Testing

Table of Contents

Introduction

This guide explains how to set up and manage automated daily test execution using GitHub Actions for the Unity Health Automation project. The workflow runs tests daily at 12:00 PM IST and sends email notifications with test results.

Prerequisites

- GitHub account with repository access
- Node.js and npm installed locally
- Gmail account for notifications
- Basic understanding of Git commands

Workflow Setup

1. Repository Structure

Ensure your repository has the following structure:

```
.github/
  └── workflows/
      └── daily-tests.yml
```

2. Workflow File

The `daily-tests.yml` file contains:

```
name: Daily Automated Tests

on:
  schedule:
    # Runs at 12:00 PM IST (06:30 UTC) every day
    - cron: '30 6 * * *'
  workflow_dispatch:  # Allows manual trigger of the workflow

jobs:
  test:
    runs-on: windows-latest
    steps:
    - uses: actions/checkout@v4
    - name: Set up Node.js
      uses: actions/setup-node@v4
      with:
        node-version: '18'
        cache: 'npm'
    # ... (other steps as defined in the workflow)
```

GitHub Configuration

1. Push Workflow File

```
# Add the workflow file
git add .github/workflows/daily-tests.yml

# Commit the changes
git commit -m "Add daily test automation workflow"

# Push to GitHub
git push origin main
```

2. Set Up GitHub Secrets

1. Go to your GitHub repository
2. Click "Settings"
3. Navigate to "Secrets and variables" → "Actions"
4. Click "New repository secret"
5. Add the following secrets:
    - `EMAIL_USERNAME`: Your Gmail address
    - `EMAIL_PASSWORD`: Gmail app password
    - `NOTIFICATION_EMAIL`: Recipient email address

Email Setup

1. Gmail App Password Generation

1. Go to Google Account settings
2. Navigate to Security
3. Enable 2-Step Verification if not enabled
4. Go to App Passwords
5. Select:
    - App: Mail
    - Device: Other (Custom name)
6. Name it "GitHub Actions"
7. Copy the 16-character password
8. Use this as your `EMAIL_PASSWORD` secret

Monitoring and Maintenance

1. Checking Workflow Status

1. Go to repository's "Actions" tab
2. Click on "Daily Automated Tests"
3. View recent runs and their status

2. Accessing Test Reports

1. Go to a specific workflow run
2. Scroll to "Artifacts" section
3. Download:
    - allure-report
    - playwright-report

3. Manual Trigger

1. Go to "Actions" tab
2. Select "Daily Automated Tests"
3. Click "Run workflow"
4. Select branch
5. Click "Run workflow"

Troubleshooting

Common Issues and Solutions

1. **Workflow Not Running**
    - Check if the cron schedule is correct
    - Verify repository permissions
    - Check GitHub Actions is enabled
2. **Email Notifications Not Received**
    - Verify Gmail app password is correct
    - Check spam folder
    - Verify email secrets are properly set
3. **Tests Failing**
    - Check test logs in workflow run
    - Verify environment variables
    - Check for dependency issues
4. **Report Generation Issues**
    - Verify Allure installation
    - Check file permissions
    - Verify artifact upload settings

Support Resources

- GitHub Actions Documentation: https://docs.github.com/en/actions
- Playwright Documentation: https://playwright.dev/docs/intro
- Allure Framework Documentation: https://docs.qameta.io/allure/

Best Practices

1. **Security**
    - Never commit sensitive data
    - Use GitHub secrets for credentials
    - Regularly rotate app passwords
2. **Maintenance**
    - Monitor workflow runs regularly
    - Update dependencies periodically
    - Keep Node.js version updated

3. **Documentation**
    - Document any workflow changes
    - Keep test documentation updated
    - Maintain changelog

Additional Resources

Useful Commands

```
# Check workflow status
gh run list

# View workflow logs
gh run view <run-id>

# Download artifacts
gh run download <run-id>
```

Recommended Tools

- GitHub CLI for workflow management
- VS Code for local development
- Postman for API testing
- Allure for report generation

Conclusion

This setup provides a robust foundation for automated testing. Regular monitoring and maintenance will ensure its continued effectiveness. For any issues or questions, refer to the troubleshooting section or contact the development team.