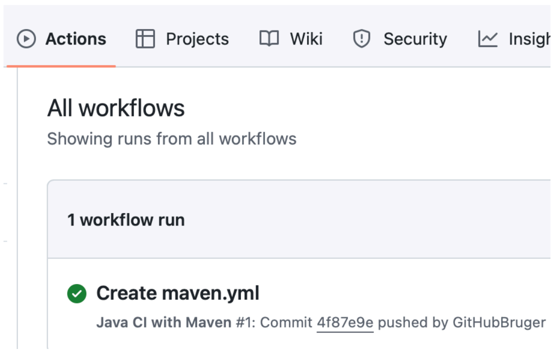
Demo: Hello GitHub Actions

1. Create a new Spring Boot application.
2. In IntelliJ, share project on GitHub (repo must be public to use Qodana Cloud later).
3. In the repository on GitHub, select the Actions tab.
4. Choose the workflow “Java with Maven” and click the Configure button.
5. Edit the maven.yml file (directly on GitHub) as shown below and commit to main.

```yaml
name: Java CI with Maven

on:
  push:
    branches: [ "main" ]
  pull_request:
    branches: [ "main" ]

jobs:
  build:

    runs-on: ubuntu-latest

    steps:
    - uses: actions/checkout@v4
    - name: Set up JDK 22
      uses: actions/setup-java@v4.3.0
      with:
        java-version: '22'
        distribution: 'temurin'
        cache: maven
    - name: Build with Maven
      run: mvn -B test
```

6. On GitHub, check that the workflow ran successfully:



All workflows
Showing runs from all workflows

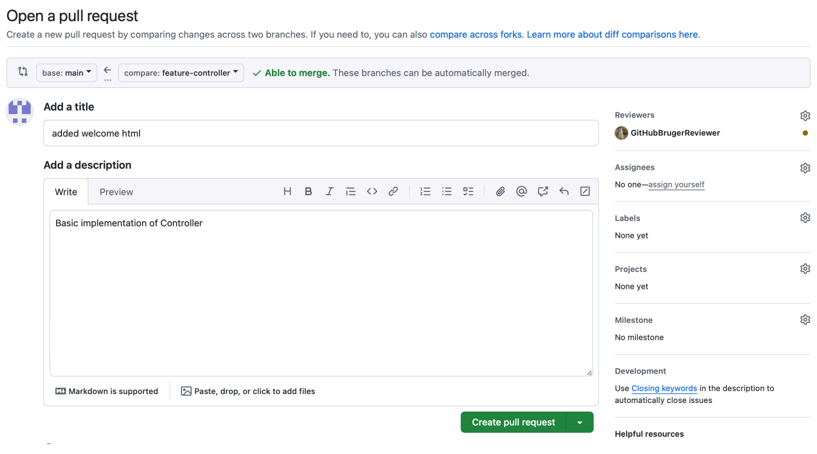
1 workflow run

Create maven.yml
Java CI with Maven #1: Commit 4f87e9e pushed by GitHubBruger

7. In IntelliJ, update the local project (Git - update)
8. Locally, create a new feature-controller branch from the main branch.
9. Create a package controller and a Spring Boot Controller class e.g. WelcomeController in this package.
10. Add a simple method which handles a GET request and returns e.g. the string “welcome”.
11. Add a welcome.html file to the templates package (the file does not need to be edited).
12. Commit the changes (locally).

Before pushing the feature-controller branch to the remote repo and making a pull request to merge the feature-controller branch into the main branch, we should ensure that no merge conflicts will occur. This means we should update the local main branch to reflect the current state of the remote main branch - our colleagues may be been busy merging their branches into the remote main. The updated local main branch should then be merged into the local feature-controller branch and any merge conflicts resolved. The feature-controller branch is now good to be pushed to the feature-controller branch on the remote repo. A pull request can then be made to merge the feature-controller branch on the remote into the main branch on the remote.

13. Update the local project.
14. Merge main into feature-controller branch. Resolve any conflicts.
15. Push feature-controller branch to the remote.
16. On GitHub, press Compare & pull request. To add a reviewer for the pull request, the reviewer must be added as a collaborator.


Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks. Learn more about diff comparisons here.

base: main ← compare: feature-controller ✓ Able to merge. These branches can be automatically merged.

Add a title

added welcome html

Add a description

Basic implementation of Controller

Reviewers: GitHubBrugerReviewer

Assignees: No one—assign yourself

Labels: None yet

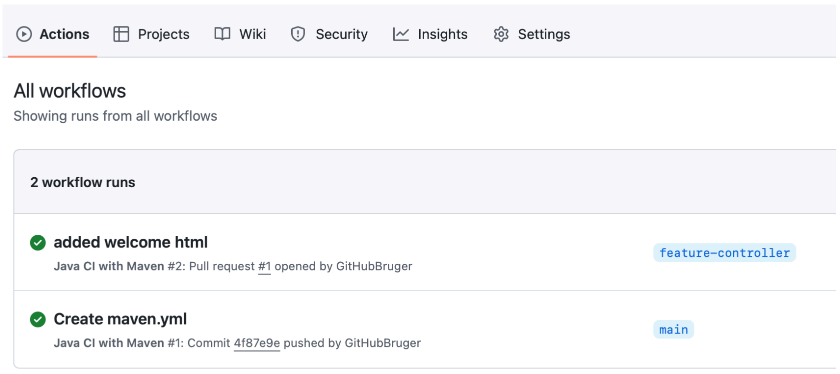
Projects: None yet

Milestone: No milestone

Development: Use Closing keywords in the description to automatically close issues

Helpful resources

17. Select Create Pull request. This triggers the GitHub Action.
18. The action has successfully completed i.e. the application was able to compile and a notification (email) has been sent to the reviewer to review the pull request.



All workflows

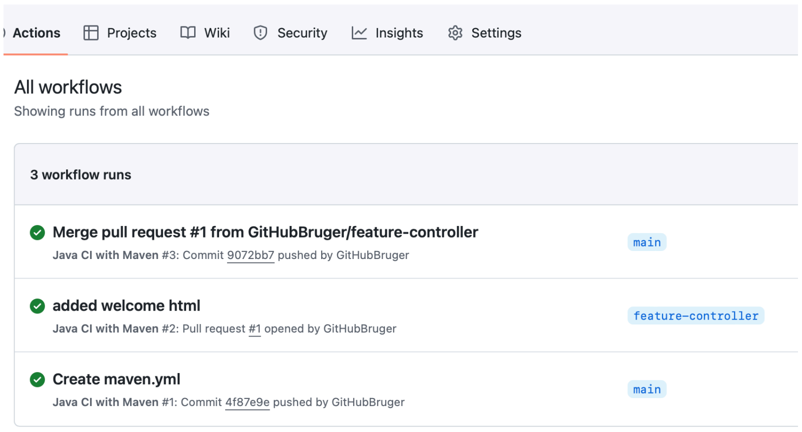
Showing runs from all workflows

2 workflow runs

✓ added welcome html — Java CI with Maven #2: Pull request #1 opened by GitHubBruger — feature-controller

✓ Create maven.yml — Java CI with Maven #1: Commit 4f87e9e pushed by GitHubBruger — main

19. The reviewer reviews the changes and approves them.
20. The developer then merges the feature branch into main and then deletes it.This also triggers the workflow.



All workflows
Showing runs from all workflows

3 workflow runs

Merge pull request #1 from GitHubBruger/feature-controller — main
Java CI with Maven #3: Commit 9072bb7 pushed by GitHubBruger

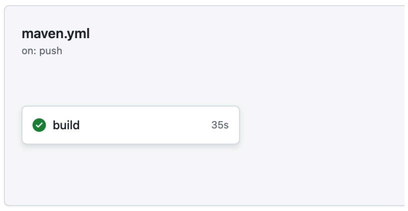
added welcome html — feature-controller
Java CI with Maven #2: Pull request #1 opened by GitHubBruger

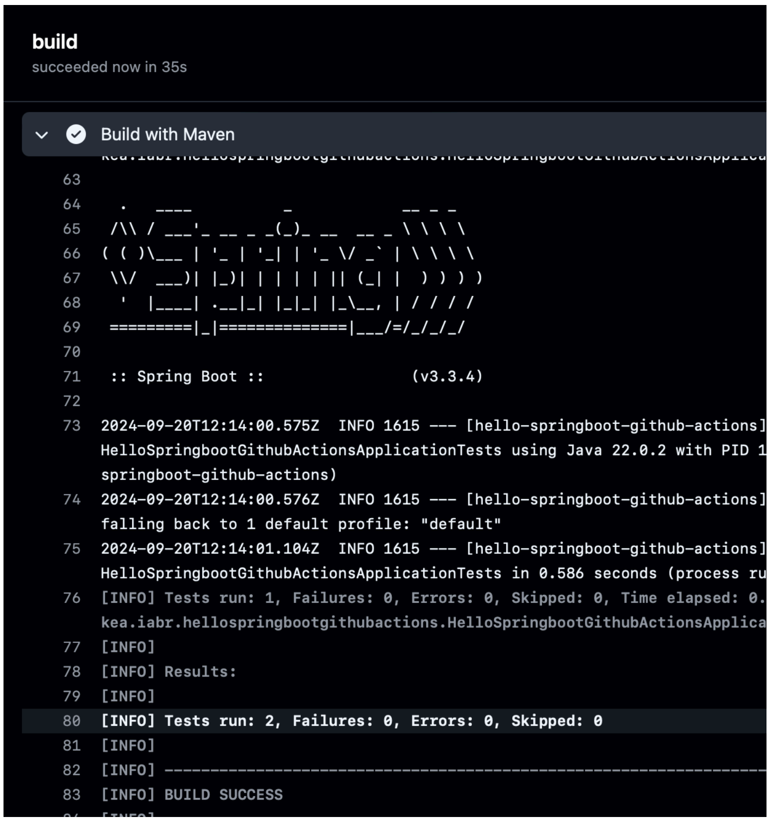
Create maven.yml — main
Java CI with Maven #1: Commit 4f87e9e pushed by GitHubBruger

21. The next step is to add unit testing. This would normally be done in the previous steps when implementing the endpoint. For illustrative purposes it is done separately.
22. Create a local feature-endpoint branch from local main branch.
23. Create a test class for the WelcomeController.
24. Implement a test method for the “welcome” endpoint.

```java
@WebMvcTest(WelcomeController.class)
class WelcomeControllerTest {

    @Autowired
    private MockMvc mockMvc;

    @BeforeEach
    void setUp() {
    }

    @AfterEach
    void tearDown() {
    }

    @Test
    public void testWelcome() throws Exception {
        mockMvc.perform(get( urlTemplate: "/welcome")) // Perform a GET request to /welcome
                .andExpect(status().isOk()) // Expect HTTP 200 OK
                .andExpect(view().name( expectedViewName: "welcome")); // Expect the view name to be "welcome"
    }
}
```

25. Run the test locally to ensure the test passes.
26. Commit the changes to the local feature-endpoint branch.
27. Update the local main branch.
28. Merge the local main into the feature-endpoint branch.
29. Push the feature-endpoint branch to the repo on GitHub.
30. Create a pull request to merge the remote feature-endpoint branch into the remote main branch. (This will trigger the workflow.)
31. Observe that the tests were run and passed by clicking the build job:

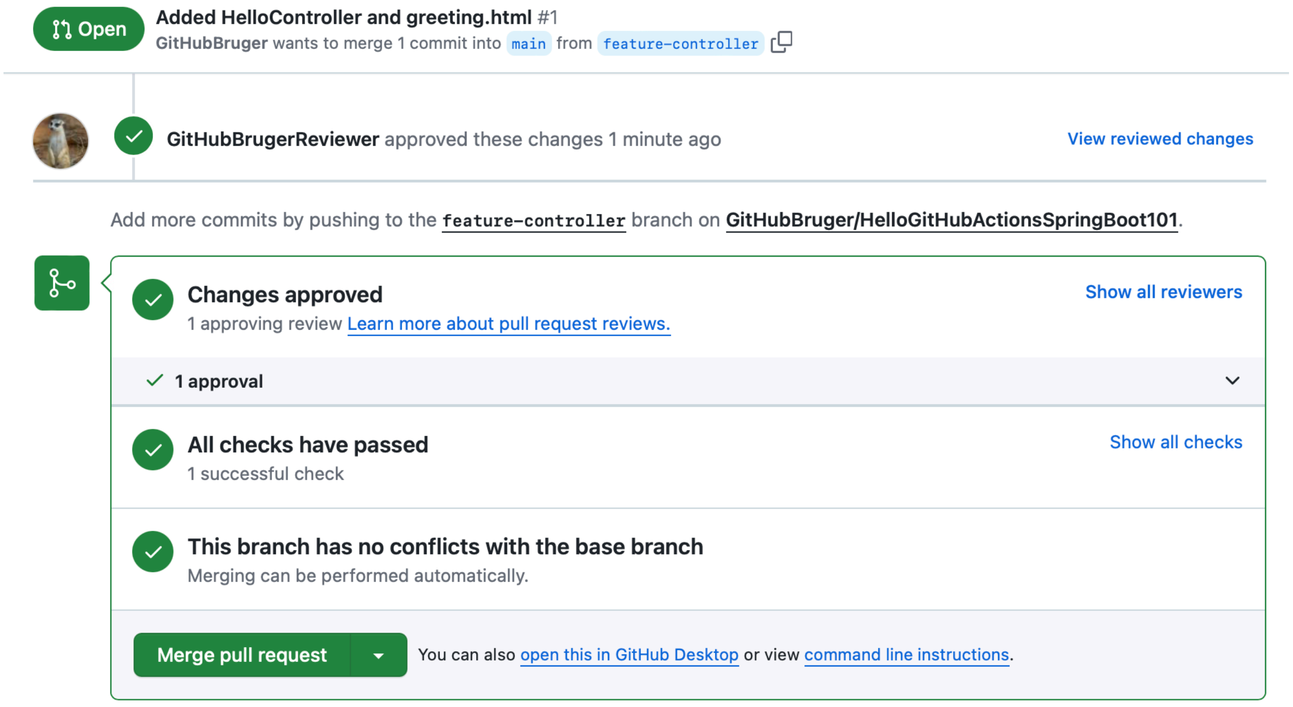



build

succeeded now in 35s

Build with Maven

```
kea.iabr.hellospringbootgithubactions.HelloSpringbootGithubActionsApplica

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/

 :: Spring Boot ::                (v3.3.4)

2024-09-20T12:14:00.575Z  INFO 1615 --- [hello-springboot-github-actions]
HelloSpringbootGithubActionsApplicationTests using Java 22.0.2 with PID 1
springboot-github-actions)
2024-09-20T12:14:00.576Z  INFO 1615 --- [hello-springboot-github-actions]
falling back to 1 default profile: "default"
2024-09-20T12:14:01.104Z  INFO 1615 --- [hello-springboot-github-actions]
HelloSpringbootGithubActionsApplicationTests in 0.586 seconds (process ru
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.
kea.iabr.hellospringbootgithubactions.HelloSpringbootGithubActionsApplica
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 2, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO]
```


Open Added HelloController and greeting.html #1

GitHubBruger wants to merge 1 commit into `main` from `feature-controller`

GitHubBrugerReviewer approved these changes 1 minute ago   View reviewed changes

Add more commits by pushing to the `feature-controller` branch on GitHubBruger/HelloGitHubActionsSpringBoot101.

Changes approved   Show all reviewers
1 approving review Learn more about pull request reviews.

✓ 1 approval

All checks have passed   Show all checks
1 successful check

This branch has no conflicts with the base branch
Merging can be performed automatically.

Merge pull request ▾ You can also open this in GitHub Desktop or view command line instructions.