**Subject: Middleware Team Position on GitHub Actions Adoption**

**TO:**

**From:** SM, RP, CA **Date:** xx.xx.xxxx **Subject:** Justification for GitHub Actions Use by the Middleware Team

**1. Introduction**

As part of our Automation and modernization efforts, the Middleware team has evaluated available CI/CD tools and determined that GitHub Actions offers significant benefits for our workflows. This document outlines our rationale, capability, and readiness to adopt GitHub Actions in a secure and scalable manner within our enterprise network.

**2. Reasons for Adopting GitHub Actions**

*A. Native GitHub Integration*

* Our code repositories are already hosted in GitHub Enterprise.
* GitHub Actions provides native CI/CD integration with GitHub features: PR events, secrets, branch protections, etc.
* Reduces reliance on external CI systems and eliminates context switching.

*B. Improved Developer Experience*

* The GitHub workflow yaml-based pipeline syntax is clean, version-controlled, and developer-friendly.
* Direct access to GitHub APIs, cloud integrations, and environment protection rules.

*C. Workflow Reusability and Modularity*

* Reusable workflows across middleware components and Java-based microservices.
* Shared internal actions can be versioned, tested, and reused securely across the org.

*D. Auditability and Compliance*

* Every GitHub Action execution is traceable with logs, commit SHAs, and signed provenance.
* Fine-grained permissions for runners, secrets, and workflows meet enterprise audit requirements.

**3. Technical Readiness of the Middleware Team**

A. Applications Managed by Middleware Team

* Our team is responsible for managing and automating infrastructure for:
  + IBM MQ (enterprise messaging and transaction coordination)
  + Apache Kafka (streaming data platform)
  + WebSphere and Liberty servers (Java application hosting)
  + Other business-critical middleware components used across multiple business units

B. Runner Infrastructure

* We have the expertise and operational experience to host self-hosted GitHub Actions runners in:
  + Kubernetes GitHub Actions Runner Controller
* Our team can ensure:
  + Autoscaling and resource isolation per workflow/team
  + On-premise runner lifecycle management
  + Network segmentation and monitoring for internal security

C. Security Expertise

* We have in-house security specialists familiar with:
  + GitHub's security model for Actions
  + Runner network isolation, vault integration for secrets.
  + Container security scanning (Trivy, Aqua, SonarQube integration)

D. Governance and Policy Enforcement

* We will enforce:
  + Only internal or validated GitHub Actions
  + Org-wide workflow permissions and secret scope restrictions
  + Role-based access controls for job triggers and environment usage

**4. Strategic Benefits to the Organization**

* Accelerates CI/CD workflows for APIs, Java services, IBM MQ, Kafka pipelines, and WebSphere/Liberty builds.
* Aligns with GitOps model and modern DevSecOps.
* Reduces overhead compared to maintaining full-scale Jenkins stacks.
* Easier onboarding for developers — no additional CI toolchain learning curve.

**Recent suggestions why GitHub Actions were not currently enabled for the entire BBB**

A recent internal note suggested that GitHub Actions are not currently used in our environment due to security, cost, and compliance concerns. The note also recommended Jenkins as an alternative. We understand Jenkins is our main CICD tool but we are not trying to replace Jenkins, we are merely trying to use a solution that is less costly. Because the Middleware team take serious regards with security, Governance and compliance, we have evaluated some of the the accuracy of those concerns that were shared in the chat and compared them to the security and operational posture of Jenkins.

**Jenkins Plugin Security Concerns:**

There is a concern that GitHub Actions plugins are risky.

I believe this is true only for public Actions can pose security risks. Can be mitigated via allowlists and trusted Actions only.

BUT, If plugin security is a blocker for GitHub Actions, Jenkins should be scrutinized equally because Jenkins has.

* Thousands ofcommunity plugins, many of which are unvetted and lack recent maintenance.
* Jenkins security advisories regularly report plugin vulnerabilities, including remote code execution.
* Plugin sandboxing is less mature compared to GitHub’s Action security features, lik verified publishers, permission scoping...

**GitHub Actions violate HIPAA**

Only if using GitHub-hosted runners. Self-hosted runners fully support HIPAA compliance when isolated. GitHub Actions can be HIPAA-compliant if using self-hosted runners within a private network, Data never leaves internal infrastructure.

**GitHub Actions "send workload to Git"**

This is not understood how and why because workflows pull source code from GitHub; no data is sent back to GitHub during execution on self-hosted runners.

**5. Recommendations**

1. Pilot a GitHub Actions implementation using self-hosted runners inside our secure network.
2. Define a curated set of approved GitHub Actions for use.
3. Use organization-level policies to restrict third-party code execution.
4. Compare maintenance overhead and developer experience between GitHub Actions and Jenkins.

**6. Conclusion**

While the concerns raised are valid in regulated environments, all of them are equally or more applicable to Jenkins. With proper governance, GitHub Actions can provide a secure, compliant, and modern CI/CD experience that aligns well with our internal goals. It is recommended we revisit its adoption through a secure, controlled proof of concept.

Note: We are not trying to replace Jenkins with GitHub runners, we are only looking to try pot a solution which we think will better suit a proper and efficiency for our team