Automation GHES

Testcase details

Testcase	Steps	Output
PersonalAccessToken **(Removed)**	Operation: - Go to setting – developers tool – personal access token - Check for token named pat_auto - If not found, create a new token named pat_auto - Tick repo checkbox - Create token Verification: - Go to setting – developers tool – personal access token - Check for token named pat_auto	Operation: - pat_auto can be created Verification: - pat_auto already existed
OrganizationBasicData	Operation: - Go to create organization page - Enter name 'basic-data'. - Complete setup Verification: - Go to organization basic-data.	Operation: - Organization basic-data can be created Verification: - Organization basic-data can be access and existed.
RepoData	Operation: - Go to organization basic-data. - Create a new repository - Enter name 'data' - Complete setup Verification: - Go to repository data inside organization basic-data	Operation: - Repo data can be created inside organization basic-data Verification: - Repo data can be accessed inside organization basic-data
SampleBranch	Operation: - Go to repo data main branch - Check for 'sample-branch-1'. - If the branch not found, create 'sample-branch-1' - Inside sample-branch-1, check for 'sample-branch-1-file.txt'. - If the file is missing, upload 'sample-branch-1-file.txt' - Go to repo data main branch - Check for 'sample-branch-2' - If the branch not found, create 'sample-branch-2' - Inside sample-branch-1, check for 'sample-branch-2-file.txt' - If the file is missing, upload 'sample-branch-2-file.txt' Verification: - Go to repo data - Access sample-branch-1	Operation: - Sample-branch-1 can be created - Sample-branch-1-file.txt can be uploaded in sample-branch-1 - Sample-branch-2 can be created - Sample-branch-2-file.txt can be uploaded Verification: - Sample-branch-1 can be accessed - Sample-branch-1-file.txt existed - Sample-branch-2 can be accessed - Sample-branch-2-file.txt existed

	- Check for sample-branch-1-file.txt - Access sample-branch-2 - Check for sample-branch-2-file.txt	
PullRequest	Operation: - Go to repo data - Access sample-branch-1 - Go to Pull Request - Create a pull request from sample-branch-1 to main branch - Merge Pull Request - Go to repo data - Access sample-branch-2 - Go to Pull Request - Create a pull request from sample-branch-2 to main branch - No need to merge Verification: - Go to repo data - Go to Pull Request - Check for closed pull request - Check for open pull request	Operation: - Pull request from sample-branch-1 can be created as closed pull request - Pull request from sample-branch-2 can be created as open pull request Verification: - Closed pull request existed - Open pull request existed
LFSData	Operation: - **LFS unable to automate** Verification: - Go to repo data - Check for '.psd' file - Open the psd file	Operation: - **LFS unable to automate** Verification: - '.psd' file existed - Lfs file can be rendered
Release	Operation: - Go to repo Data - Create a new release - Enter tag name 'Sample-Release-Tag' - Enter release title 'Sample Release Name' - Enter release description 'Sample Release Description' - Upload 'test.ghl' as binaries file - Publish release Verification: - Go to repo Data - Go to release - Check for sample release	Operation: - Sample release can be created and published Verification: - Sample release existed
Package	Operation: - Go to repo data - Go to actions - Search for 'Publish Node.js Package to GitHub Packages' - Configure workflow that contain 'Publish Node.js Package to GitHub Packages' - Commit workflow - **Package data unable to automate** Verification: - Go to repo data - Check for .yml file - Check for package data	Operation: - Actions is enabled - Workflow can be configured and committed Verification: - Package workflow existed - Package data existed
Issue	Operation: - Go to repo data - Go to Issues - Create new issue	Operation: - Sample open issue can be created

	- Enter name 'Sample Open Issue' - Enter description 'Sample Open Issue Description' - Upload 'sample_image.png' in description. - Submit issue - Create new issue - Enter name 'Sample Close Issue' - Enter description 'Sample Close Issue' - Upload 'sample_image.png' in description. - Submit issue - Comment 'Closing Issue' - Upload 'sample_image.png' in comment. - Close issue. Verification: - Go to repo data - Go to Issues - Check open issue - Check close issue	- Image can be uploaded to issue - Sample close issue can be opened, commented and closed Verification: - Open issue existed - Close issue existed
Project	Operation: - Go to repo data - Go to Projects - Go to Projects (Classic) - Create new Classic project - Enter project name 'Sample Open Project' - Enter project description 'Sample Open Project Description' - Choose template 'Basic kanban' - Create project - Create new Classic project - Enter project name 'Sample Close Project' - Enter project description 'Sample Close Project Description' - Choose template 'Basic kanban' - Create project - Close project Verification: - Go to repo data - Go to Projects - Check sample open project - Check sample close project	Operation: - Sample projects can be created - Sample project can be closed. Verification: - Sample open project exists. - Sample close project exists.
Wiki	Operation: - Go to repo data - Go to Wiki - Create first page - Enter Wiki title, description, upload sample image and message. - Save page - Create second page - Enter Wiki title, description, upload sample image and message - Save page Verification: - Go to repo data - Go to Wiki	Operation: - Wiki first page can be created - Wiki second page can be created Verification: - Wiki first page exists - Wiki second page exists

	- Check for wiki first page - Check for wiki second page	
Webhook	Operation: - Go to repo data - Go to Settings - Go to Hooks - Add Webhook - Enter webhook details - Tick second checkbox (Send me everything) - Add Webhook Verification: - Go to repo data - Go to Settings - Go to Hooks - Check for webhook	Operation: - Webhook can be created Verification: - Webhook exists
PreReceiveHook **(Removed)**	Operation: - Go to repo data - Upload 'pre-receive-hook.rb' file - Go to enterprise setting - Go to Settings - Go to Hooks - Add pre-receive hook - Enable all except second checkbox Verification: - Go to repo data - Check for 'pre-receive-hook.rb' file - Go to enterprise setting - Go to Settings - Go to Hooks - Check for pre-receive hook - Open pre-receive hook - Enable second checkbox and save - Go to repo data - Upload and commit any file - Check for hook triggers - Go to enterprise setting - Go to Settings - Go to Hooks - Open pre-receive hook - Disable second checkbox and save	Operation: - Pre-receive-hook.rb file can be uploaded - Pre-receive hook can be created Verification: - Pre-receive-hook.rb file exists - Second checkbox has been enabled. - Pre-receive hook triggers upon commit file - Second checkbox has been disabled.
RepoCodeScanning	Operation: - Go to organization basic-data - Create new repo 'code-scanning' - Go to repo code-scanning - Upload 'index.js' file - Go to repo setting - Enable advance setting - Go to actions - Create codeql workflows - **Runner cannot be automated** Verification: - Go to organization basic-data - Go to repo code-scanning - Check for index.js - Go to repo setting - Check advance setting - Go to code - Check for qodeql workflows	Operation: - Repo code-scanning can be created - 'index.js' can be uploaded - Advance setting can be enabled - Codeql workflows can be created Verification: - Repo code-scanning can be accessed - 'index.js' exists - Advance setting enabled - Codeql workflows existed - Runners existed

	- Go to organization basic-data setting - Go to actions - Go to Runners	
OrganizationSecurityAlerts	Operation: - Go to Create Organization Page - Enter name 'security-alerts' - Complete setup Verification: - Go to organization security-alerts	Operation: - Organization 'security-alerts' can be created and accessed Verification: - Organization 'security-alerts' existed and can be accessed
RepoDependabot	Operation: - Go to organization security-alerts - Create repo 'dependabot' - Go to repo setting - Go to Code security and analysis - Enable dependabot alerts - **GIthub Connect unable to automate due to security reason** - **Dependabot files unable to automate** Verification: - Go to repo dependabot inside organization security-alerts - Go to repo setting - Check for dependabot setting - Go to Code - Check dependabot files	Operation: - Repo dependabot can be created and accessed. - Dependabot setting can be enabled Verification: - Repo dependabot exist and can be accessed - Dependabot setting has been enabled - Dependabot files exists
RepoSecretScanning	Operation: - Go to organization security-alerts - Create repo 'secret-scanning' - Go to repo setting - Go to Code security and analysis - Enable Advance setting - Enable Secret Scanning - Add secret pattern - Enter secret pattern details - Publish pattern - Go to code - Upload file containing secret pattern Verification: - Go to organization security-alerts - Go to repo secret-scanning - Go to repo setting tab - Go to Code security and analysis - Check advance setting status - Check secret scanning status - Check pattern status - Go to Security tab - Check secret scanning alerts	Operation: - Repo secret-scanning can be created and accessed - Advance setting can be enabled - Secret scanning setting can be enabled - Pattern can be created and can be published - Secret pattern file can be uploaded Verification: - Repo secret-scanning exist and accessible - Advance security enabled - Secret scanning setting enabled - Pattern created and published - Secret scanning file exist - Secret scanning alerts appears
OrganizationRunner **(Removed)**	Add organization for runners	Verify organization for runners exist
RepoRunner **(Removed)**	Add repo for repo-level runner Repo-level runner unable to automate	Verify repo for runner exist. Verify repo-level runner exist.
OrgRunner **(Removed)**	Org-level runner unable to automate	Verify org-level runner exist
EnterpriseRunner **(Removed)**	Enterprise-level runner unable to automate	Verify enterprise-level runner exist.
OrganizationEmpty	Add empty organization	Verify empty organization exist

(Removed)		
Gist	Operation: - Go to Create Gist page - Enter Gist description 'Secret Gist Data - Enter Gist filename 'secret_gist.md' - Enter Gist details - Create secret gist - Go to Create Gist page - Enter Gist description 'Public Gist Data' - Enter Gist filename 'public_gist.md' - Enter Gist details - Create public gist Verification: - Go to Gist page - Check for secret gist - Check for public gist	Operation: - Secret gist can be created - Public gist can be created Verification: - Gist page can be accessed - Secret gist exists - Public gist exists
NewUser	Operation: - Go to Site Admin - Go to Invite User - Enter username 'user1' - Enter email 'user1@ghe-test.net' - Send invitation - Open invitation link given on new tab - Change password to 'Test1234' - Close tab - Go to repo data in organization basic-data - Go to setting tab - Go to Collaborators and teams - Add user1 and save Verification: - Go to repo data in organization basic-data - Go to setting tab - Go to Collaborators and teams - Check for user1 - Sign out current user - Login user1 - Username 'user1' - Password 'Test1234' - Access repo data inside organization basic-data	Operation: - User1 can be invited - Invitation link received can be opened - User1 can be added as collaborator Verification: - User1 is a collaborator for repo data - User1 can be logged in
TestEmail	Operation: - Sign out current user - Login user1 - Username 'user1' - Password 'Test1234' - Access repo data - Go to Issues tab - Open Sample Open Issue - Comment on the issue - Sign out user1 Verification: - Go to repo data - Go to Issues tab	Operation: - User1 can be logged in - User1 can accessed repo data - User1 can comment on issue Verification: - Sample open issue has comment from user1

	- Open Sample Open Issue - Check comment on issue	
ManagementConsoleSetting	Operation: - **Not automated** Verification: - Go to management console setting - Check for email - Check fo actions - Check for packages - Check for dependency graph - Check for chat integration	Operation: - **Not automated** Verification: - Management console setting can be accessed - Email enabled - Actions enabled - Packages enabled - Dependency graph enabled - Chat integration enabled