**Migration Operations Plan (MOP) Bitbucket to GitHub Repository Migration**

## **1. Pre-Migration Planning**

### **1.1 Access Control and Permissions**

* **Step 1:** Log in to both your Bitbucket and GitHub accounts.
* **Step 2:** Ensure that you have admin access to all repositories that you are migrating from Bitbucket.
  + For Bitbucket: Go to Repository Settings > User and Group Access.
  + For GitHub: Navigate to Settings > Collaborators & Teams to check your permissions.
* **Step 3**: Ensure GitHub permissions are properly mapped for users. If a user in Bitbucket has write access, grant them the write role in GitHub, etc.

### **1.2 Tools and Dependencies**

* **Step 1**: Install Git on your machine by following the official guide [here](https://git-scm.com/book/en/v2/Getting-Started-Installing-Git).
  + Verify installation by running git --version in your terminal.
* **Step 2**: (Optional) Install GitHub CLI by following [this guide](https://cli.github.com/manual/).
  + Verify installation by running gh --version in your terminal.
* **Step 3**: Identify any third-party integrations (e.g., Jira, Slack) and ensure these are available on GitHub (refer to GitHub Marketplace).

### **1.3 Backup Strategy**

* **Step 1:** Log into Bitbucket and manually export each repository or use the following Git commands to clone each repository.

****

* **Step 2:** Save exported repositories and pipeline configuration files (bitbucket-pipelines.yml) in a backup folder.

## **2. Repository Migration**

### **2.1 Clone Bitbucket Repository**

* **Step 1:** Open your terminal or command line tool.
* **Step 2:** Clone the Bitbucket repository using the following command:

****

“This command clones the repository with all branches, tags, and history.”

### **2.2 Create Repository on GitHub**

* **Step 1**: Log into GitHub and navigate to your organization or user account.
* **Step 2**: Click on the **New repository** button to create a new repository.
* **Step 3**: Choose a repository name and ensure it matches the original Bitbucket repo name (optional).
* **Step 4**: Set the repository to **private** or **public** based on your original Bitbucket settings.
* **Step 5**: Click **Create repository**.

### **2.3 Push Repository to GitHub**

* **Step 1**: In your terminal, change directory to the cloned Bitbucket repository:



* **Step 2**: Update the Git remote to point to GitHub:



* **Step 3**: Push the cloned repository to GitHub:



* **Step 4**: After the push is complete, visit your GitHub repository and check that all branches, commits, and tags are successfully transferred.

### **2.4 Verify Repository Migration**

* **Step 1**: Check that all branches, commits, and tags are available.
  + Use git branch -a to list all branches locally and remotely.
* **Step 2**: Compare the repository’s commit history in GitHub with the original Bitbucket repository to ensure accuracy.

## **3. Pipeline Migration (Bitbucket Pipelines to GitHub Actions)**

### **3.1 Analyze Existing Bitbucket Pipeline**

* **Step 1:** Open the bitbucket-pipelines.yml file in the root directory of the Bitbucket repository.
* **Step 2**: Identify key parts of the pipeline:
  + Build, test, and deploy steps
  + Commands executed in each step
  + Any custom environment variables or artifacts

### **3.2 Map Bitbucket Pipelines to GitHub Actions**

* **Step 1:** Understand GitHub Actions format. GitHub Actions workflows are stored in .github/workflows/ and run on specific events (e.g., push to a branch).
* **Step 2:** Translate your Bitbucket pipeline into a GitHub Actions workflow. Here’s a sample translation:

### **3.3 Create GitHub Actions Workflow**

* **Step 1**: In the GitHub repository, navigate to Code and create a .github folder in the root directory if it doesn’t already exist.
* **Step 2:** Inside the .github folder, create a workflows subfolder.
* **Step 3**: Add a new file with a descriptive name (e.g., ci.yml) and paste your GitHub Actions configuration.
* **Step 4:** Save and commit the changes.

### **3.4 Migrate Secrets and Environment Variables**

* **Step 1**: In Bitbucket, export the environment variables used in the pipeline.
* **Step 2**: Navigate to **Settings > Secrets and variables > Actions** in your GitHub repository.
* **Step 3**: Add each environment variable manually by clicking **New repository secret**.

## **4. Integration Migration (Webhooks & External Services)**

### **4.1 Replicate Webhooks**

* **Step 1:** Identify any webhooks in Bitbucket by going to Repository Settings > Webhooks.
* **Step 2:** In GitHub, navigate to Settings > Webhooks.
* **Step 3:** Add new webhooks, specifying the necessary URL and events that trigger the webhook (e.g., push, pull request).

### **4.2 Check 3rd Party Integrations**

* **Step 1**: Review third-party services (e.g., Jira, Slack) integrated in Bitbucket.
* **Step 2**: Check if equivalent integrations exist in GitHub by searching the [GitHub Marketplace](https://github.com/marketplace).
* **Step 3**: Install the required integrations and configure them as needed.

## **5. Post-Migration Validation**

### **5.1 Repository Verification**

* **Step 1:** Verify that the GitHub repository contains all branches, commits, and tags.
* **Step 2:** Compare the Git history with the Bitbucket repository to ensure accuracy.

### **5.2 Pipeline Testing**

* **Step 1:** Test your GitHub Actions pipeline by pushing changes and monitoring the results in the Actions tab.
* **Step 2**: Ensure that all steps (build, test, deploy) are executed as expected.

### **5.3 Permissions and Access Control**

* **Step 1:** Go to **Settings > Collaborators & Teams in GitHub.**
* **Step 2:** Verify that the correct permissions have been assigned to users, and branch protection rules (if any) are applied.

### **5.4 Backup and Documentation**

* **Step 1:** Store a backup of the original Bitbucket repository in a secure location.
* **Step 2:** Document all migration steps and any changes to repository or pipeline structure for future reference.

## **6. Optional Steps: Redirect and Decommission Bitbucket**

#### **6.1 Repository Redirect**

* **Step 1:** Update the README file in the Bitbucket repository to inform users that the repository has been migrated to GitHub.
* **Step 2:** Optionally, include a redirect link in the README file that points to the new GitHub repository.

#### **6.2 Decommission Bitbucket**

* **Step 1:** Once the GitHub repository and pipelines are verified and fully functional, consider deactivating or archiving the Bitbucket repository to prevent further usage.
* **Step 2:** Communicate the decommissioning plan with your team, making sure everyone is aware of the migration and the new repository location on GitHub.

## **7. Communication and Training**

* **Step 1**: Inform your team about the migration to GitHub and provide them with the new repository URLs.
* **Step 2**: Provide training or documentation on using GitHub Actions if needed, highlighting