NAME:MIDHUNA R

ROLL NO:22BCS063

Experiment No. 4

Title: Implement BitBucket Operations using Git.

Objective:

The objective of this experiment is to guide you through the process of using Git commands to interact with Bitbucket, from creating a repository to collaborating with others through pull requests.

Introduction:

Bitbucket is a web-based platform designed to provide version control, source code management, and collaboration tools for software development projects. It is widely used by teams and individuals to track changes in code, collaborate on projects, and streamline the development process. Bitbucket offers Git and Mercurial as version control systems and provides features to support code collaboration, continuous integration/continuous deployment (CI/CD), and project management.

Key Features of Bitbucket:

- Version Control: Bitbucket supports both Git and Mercurial version control systems, allowing developers to track changes, manage code history, and work collaboratively on projects.
- Repositories: In Bitbucket, a repository is a container for code, documentation, and other project assets. It houses different branches, tags, and commits that represent different versions of the project.
- Collaboration: Bitbucket enables team collaboration through features like pull requests, code reviews, inline commenting, and team permissions. These tools help streamline the process of merging code changes.
- Pull Requests: Pull requests in Bitbucket allow developers to propose and review code changes before they are merged into the main codebase. This process helps ensure code quality and encourages collaboration.
- Code Review: Bitbucket provides tools for efficient code review, allowing team members to comment on specific lines of code and discuss changes within the context of the code itself.
- Continuous Integration/Continuous Deployment (CI/CD): Bitbucket integrates with CI/CD pipelines, automating processes such as building, testing, and deploying code changes to various environments.
- Project Management: Bitbucket offers project boards and issue tracking to help manage tasks, track progress, and plan project milestones effectively.
- Bitbucket Pipelines: This feature allows teams to define and automate CI/CD pipelines directly within Bitbucket, ensuring code quality and rapid delivery.

- Access Control and Permissions: Bitbucket allows administrators to define user roles, permissions, and access control settings to ensure the security of repositories and project assets.

Benefits of Using Bitbucket:

- Version Control: Bitbucket's integration with Git and Mercurial provides efficient version control and code history tracking.
- Collaboration: The platform's collaboration tools, including pull requests and code reviews, improve code quality and facilitate team interaction.
- CI/CD Integration: Bitbucket's integration with CI/CD pipelines automates testing and deployment, resulting in faster and more reliable software delivery.
- Project Management: Bitbucket's project management features help teams organize tasks, track progress, and manage milestones.
- Flexibility: Bitbucket offers both cloud-based and self-hosted options, providing flexibility to choose the deployment method that suits the organization's needs.
- Integration: Bitbucket integrates with various third-party tools, services, and extensions, enhancing its functionality and extending its capabilities.

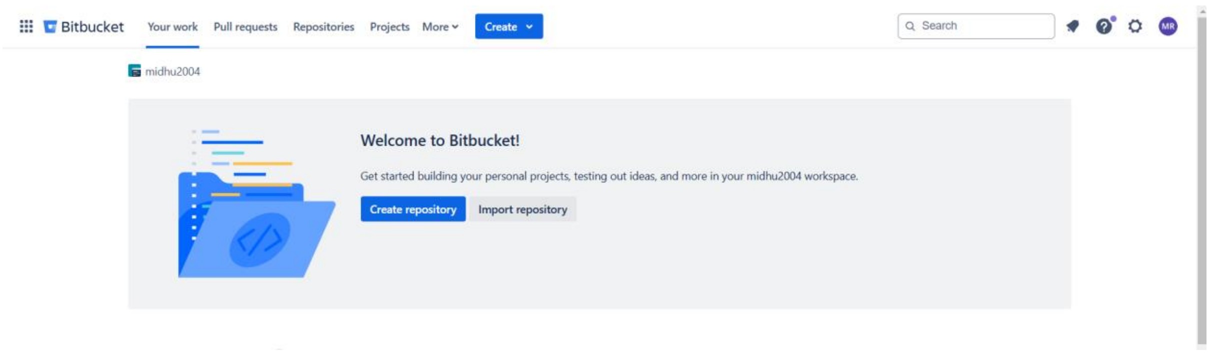
Materials:

- Computer with Git installed (https://git-scm.com/downloads)
- Bitbucket account (https://bitbucket.org/)
- Internet connection

Experiment Steps:

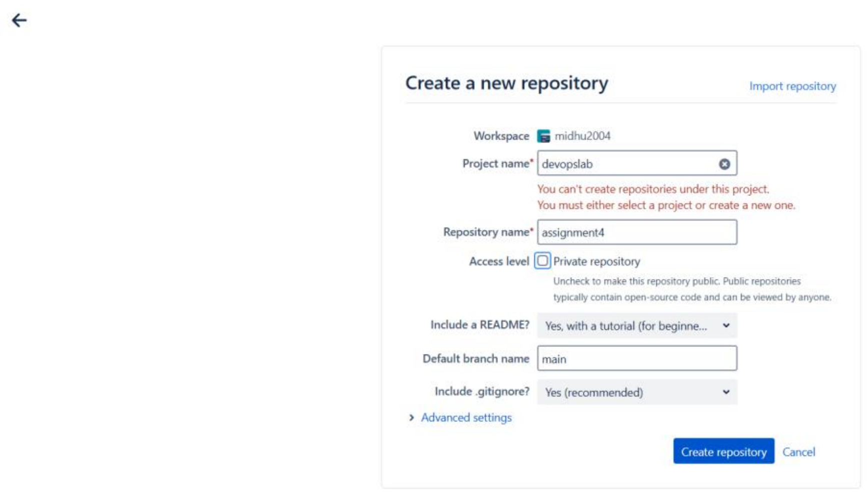
Step 1: Creating a Repository

- Sign in to your Bitbucket account.
- Click the "Create" button to create a new repository.
- Choose a repository name, visibility (public or private), and other settings.
- Click "Create repository."



Welcome to Bitbucket!

Get started building your personal projects, testing out ideas, and more in your midhu2004 workspace.


Create a new repository

Import repository

Workspace: midhu2004

Project name*: devopslab

You can't create repositories under this project.
You must either select a project or create a new one.

Repository name*: assignment4

Access level: Private repository

Uncheck to make this repository public. Public repositories typically contain open-source code and can be viewed by anyone.

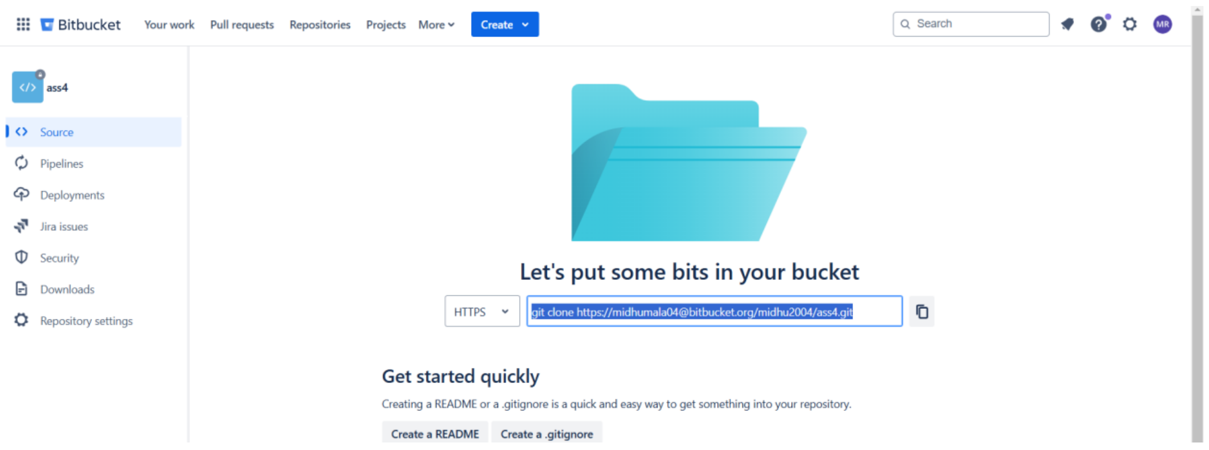
Include a README?: Yes, with a tutorial (for beginne...

Default branch name: main

Include .gitignore?: Yes (recommended)

Step 2: Cloning a Repository

- Open your terminal or command prompt.
- Navigate to the directory where you want to clone the repository.
- Copy the repository URL from Bitbucket.
- Run the following command:



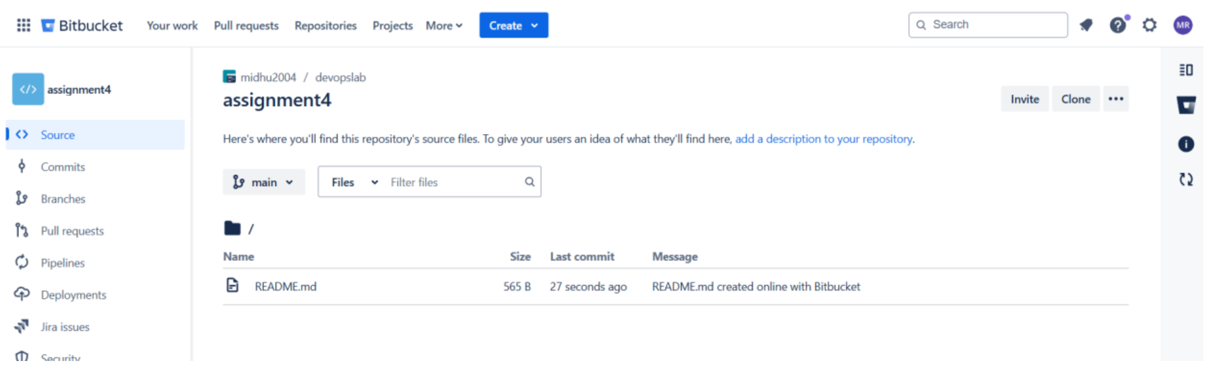
Let's put some bits in your bucket

HTTPS

```
git clone https://midhumala04@bitbucket.org/midhu2004/ass4.git
```

Get started quickly

Creating a README or a .gitignore is a quick and easy way to get something into your repository.



midhu2004 / devopslab

assignment4

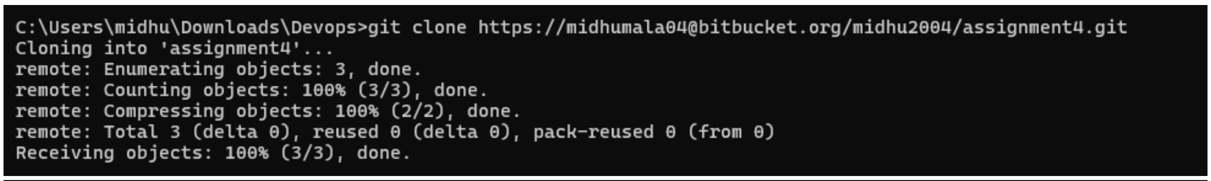
Here's where you'll find this repository's source files. To give your users an idea of what they'll find here, add a description to your repository.

/

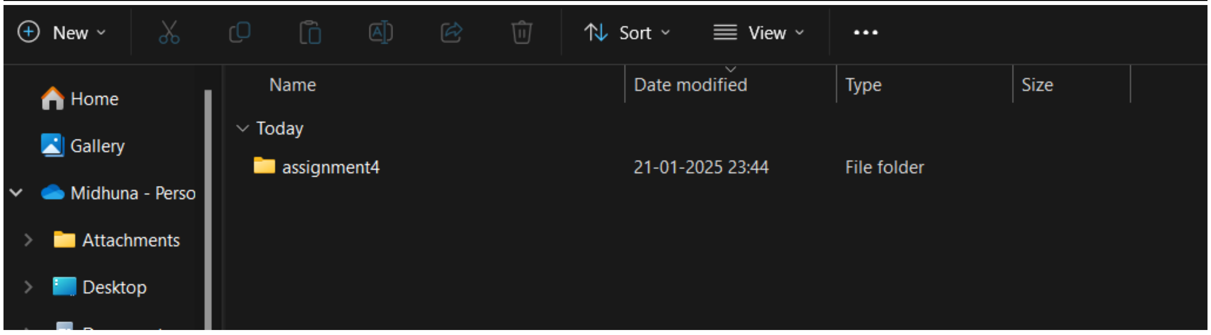
Name	Size	Last commit	Message
README.md	565 B	27 seconds ago	README.md created online with Bitbucket

git clone <repository_url>

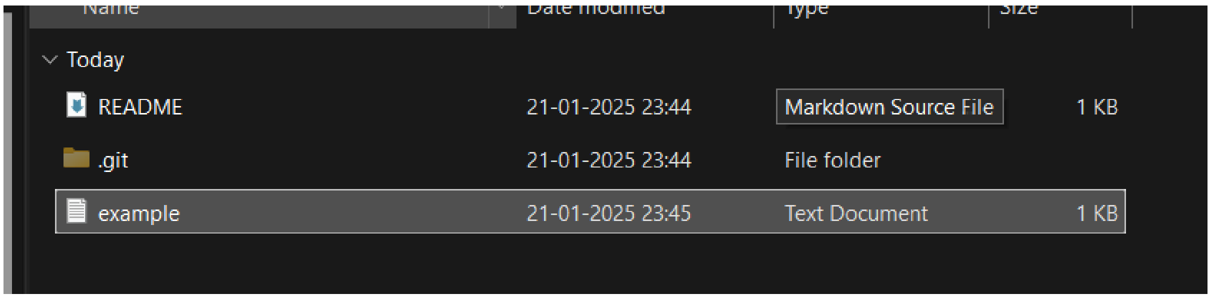
- Replace <repository_url> with the URL you copied from Bitbucket.
- This will clone the repository to your local machine.



```
C:\Users\midhu\Downloads\Devops>git clone https://midhumala04@bitbucket.org/midhu2004/assignment4.git
Cloning into 'assignment4'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
```



Name	Date modified	Type	Size
Today			
assignment4	21-01-2025 23:44	File folder	



Name	Date modified	Type	Size
Today			
README	21-01-2025 23:44	Markdown Source File	1 KB
.git	21-01-2025 23:44	File folder	
example	21-01-2025 23:45	Text Document	1 KB

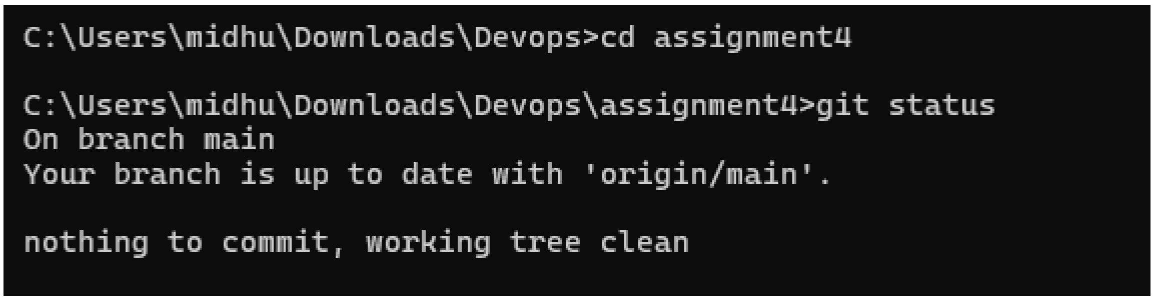
Step 3: Making Changes and Creating a Branch

- Navigate into the cloned repository:

cd <repository_name>

- Create a new text file named "example.txt" using a text editor.
- Add some content to the "example.txt" file.content:hi.
- Save the file and return to the command line.
- Check the status of the repository:

git status



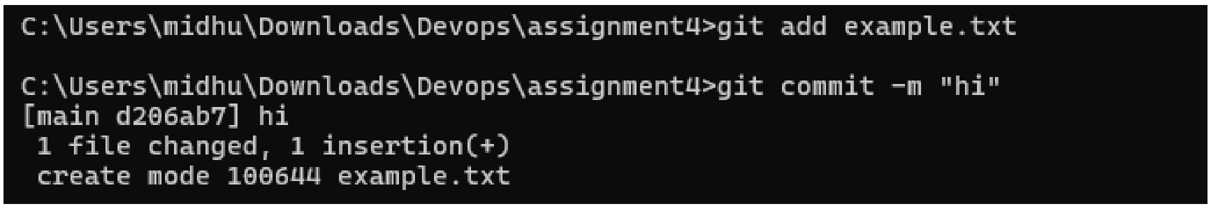
```
C:\Users\midhu\Downloads\Devops>cd assignment4

C:\Users\midhu\Downloads\Devops\assignment4>git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
```

- Stage the changes for commit:

git add example.txt

- Commit the changes with a descriptive message:

git commit -m "Add content to example.txt"



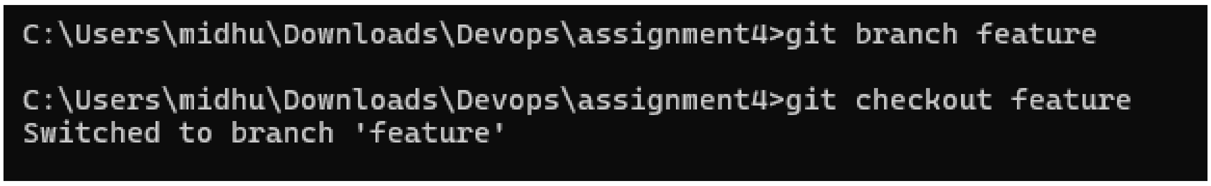
```
C:\Users\midhu\Downloads\Devops\assignment4>git add example.txt

C:\Users\midhu\Downloads\Devops\assignment4>git commit -m "hi"
[main d206ab7] hi
 1 file changed, 1 insertion(+)
 create mode 100644 example.txt
```

- Create a new branch named "feature":

git branch feature

- Switch to the "feature" branch:

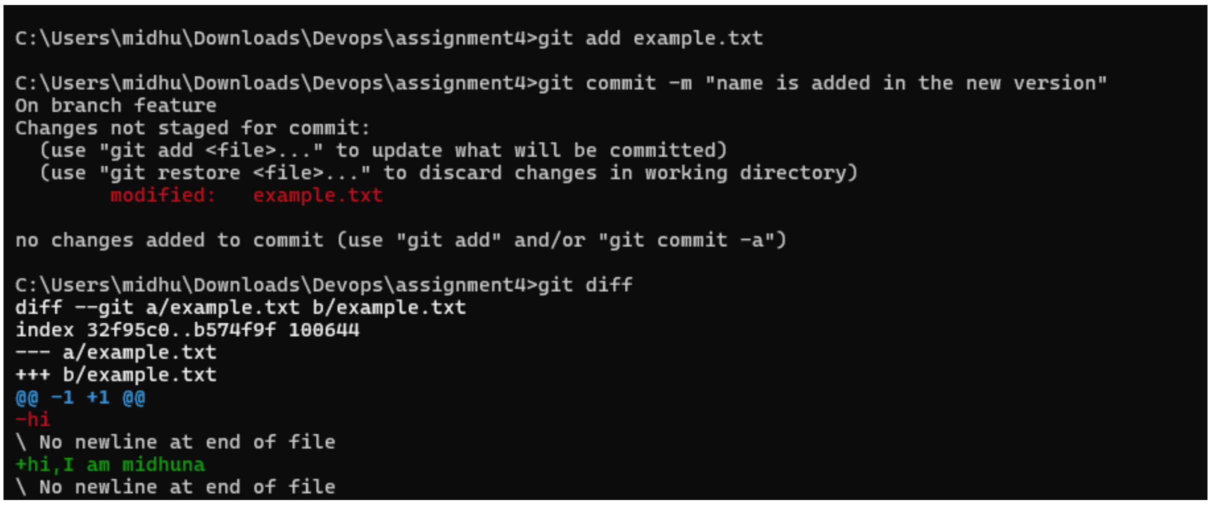
git checkout feature



```
C:\Users\midhu\Downloads\Devops\assignment4>git branch feature

C:\Users\midhu\Downloads\Devops\assignment4>git checkout feature
Switched to branch 'feature'
```

Step 4: Pushing Changes to Bitbucket

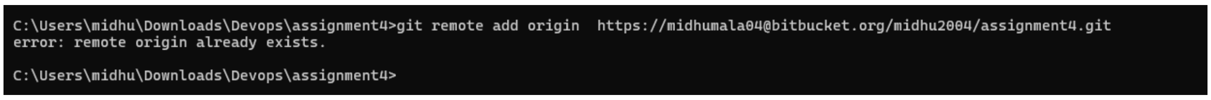
- Add Repository URL in a variable



```
C:\Users\midhu\Downloads\Devops\assignment4>git add example.txt

C:\Users\midhu\Downloads\Devops\assignment4>git commit -m "name is added in the new version"
On branch feature
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   example.txt

no changes added to commit (use "git add" and/or "git commit -a")

C:\Users\midhu\Downloads\Devops\assignment4>git diff
diff --git a/example.txt b/example.txt
index 32f95c0..b574f9f 100644
--- a/example.txt
+++ b/example.txt
@@ -1 +1 @@
-hi
\ No newline at end of file
+hi,I am midhuna
\ No newline at end of file
```

git remote add origin <repository_url>

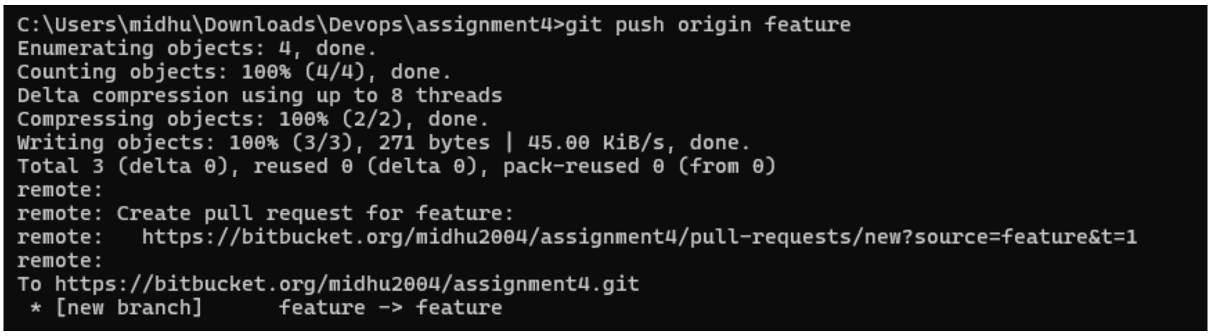
- Replace <repository_url> with the URL you copied from Bitbucket.
- Push the "feature" branch to Bitbucket:



```
C:\Users\midhu\Downloads\Devops\assignment4>git remote add origin  https://midhumala04@bitbucket.org/midhu2004/assignment4.git
error: remote origin already exists.

C:\Users\midhu\Downloads\Devops\assignment4>
```

git push origin feature

- Check your Bitbucket repository to confirm that the new branch "feature" is available.

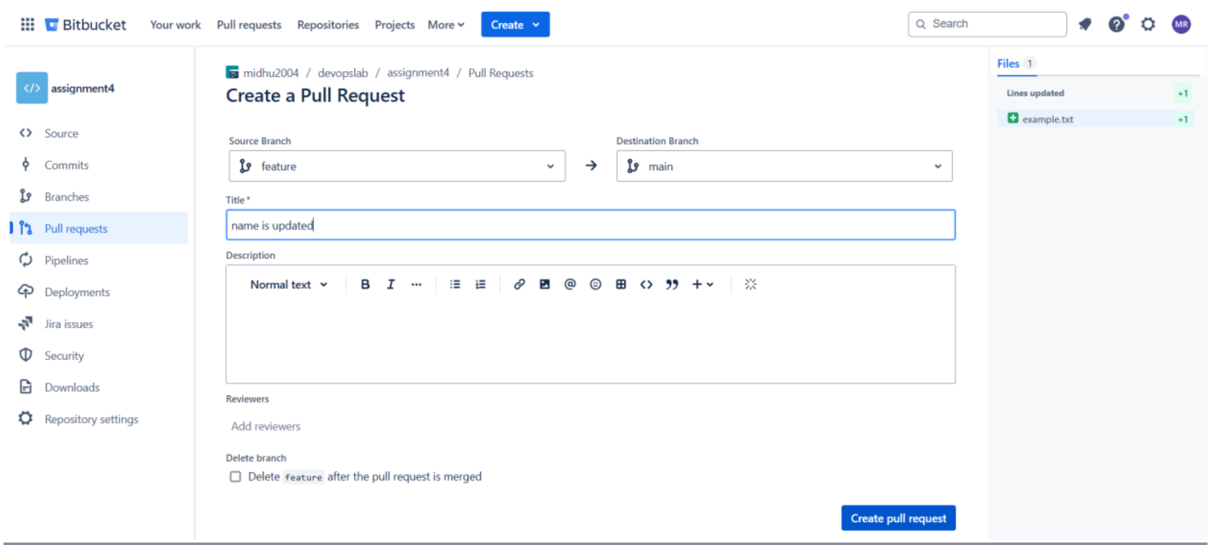


```
C:\Users\midhu\Downloads\Devops\assignment4>git push origin feature
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 271 bytes | 45.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create pull request for feature:
remote:   https://bitbucket.org/midhu2004/assignment4/pull-requests/new?source=feature&t=1
remote:
To https://bitbucket.org/midhu2004/assignment4.git
 * [new branch]      feature -> feature
```

Step 5: Collaborating through Pull Requests

1. Create a pull request on Bitbucket:

- Go to the repository on Bitbucket.
- Click on "Create pull request."
- Choose the source branch ("feature") and the target branch ("main" or "master").
- Review the changes and click "Create pull request."



midhu2004 / devopslab / assignment4 / Pull Requests

Create a Pull Request

Source Branch: feature → Destination Branch: main

Title *: name is updated

Description

Reviewers: Add reviewers

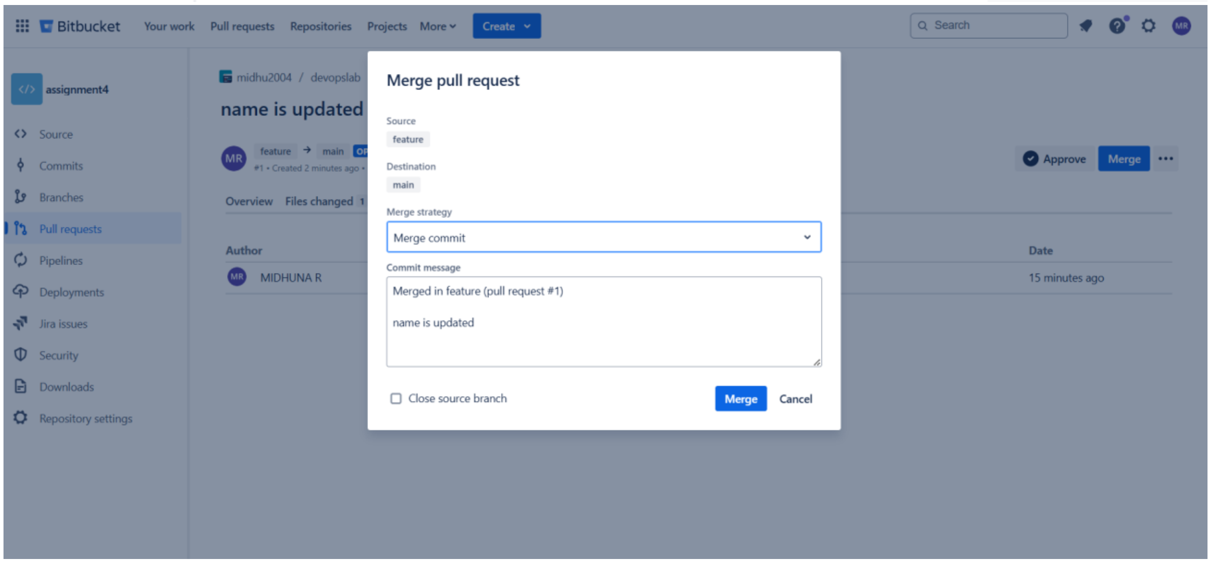
Delete branch

☐ Delete `feature` after the pull request is merged

Files 1

Lines updated +1

example.txt +1



midhu2004 / devopslab

name is updated

feature → main

#1 • Created 2 minutes ago •

Author: MIDHUNA R

Date: 15 minutes ago

Merge pull request

Source: feature

Destination: main

Merge strategy: Merge commit

Commit message:

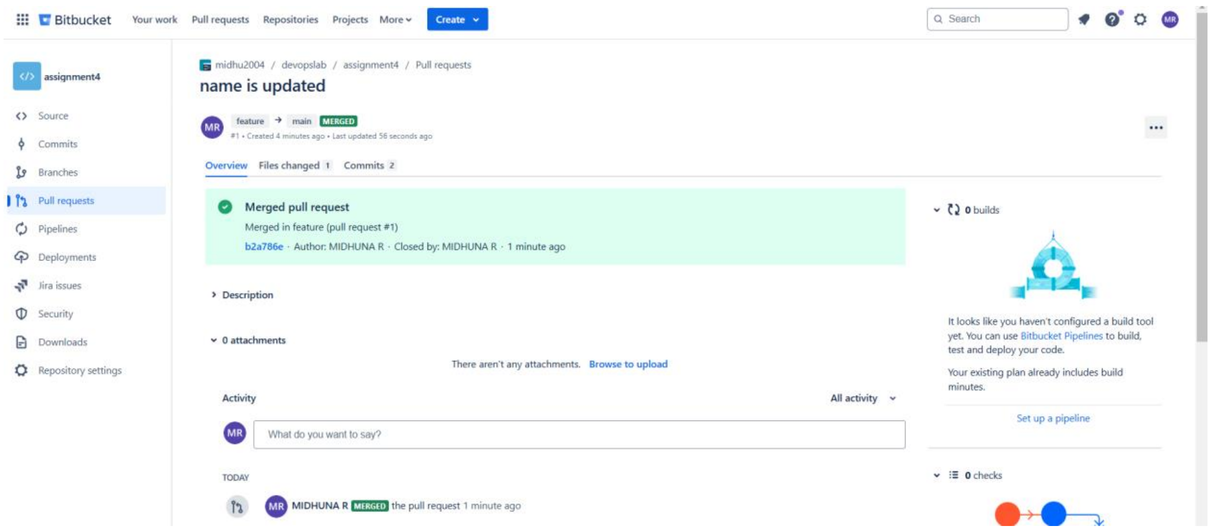
Merged in feature (pull request #1)

name is updated

☐ Close source branch

2. Review and merge the pull request:

   - Add a title and description for the pull request.
   - Assign reviewers if needed.
   - Once the pull request is approved, merge it into the target branch


Bitbucket Your work Pull requests Repositories Projects More Create Search

assignment4
- Source
- Commits
- Branches
- Pull requests
- Pipelines
- Deployments
- Jira issues
- Security
- Downloads
- Repository settings

midhu2004 / devopslab / assignment4 / Pull requests

name is updated

MR feature → main MERGED
#1 • Created 4 minutes ago • Last updated 56 seconds ago

Overview Files changed 1 Commits 2

Merged pull request
Merged in feature (pull request #1)
b2a786e · Author: MIDHUNA R · Closed by: MIDHUNA R · 1 minute ago

Description

0 attachments
There aren't any attachments. Browse to upload

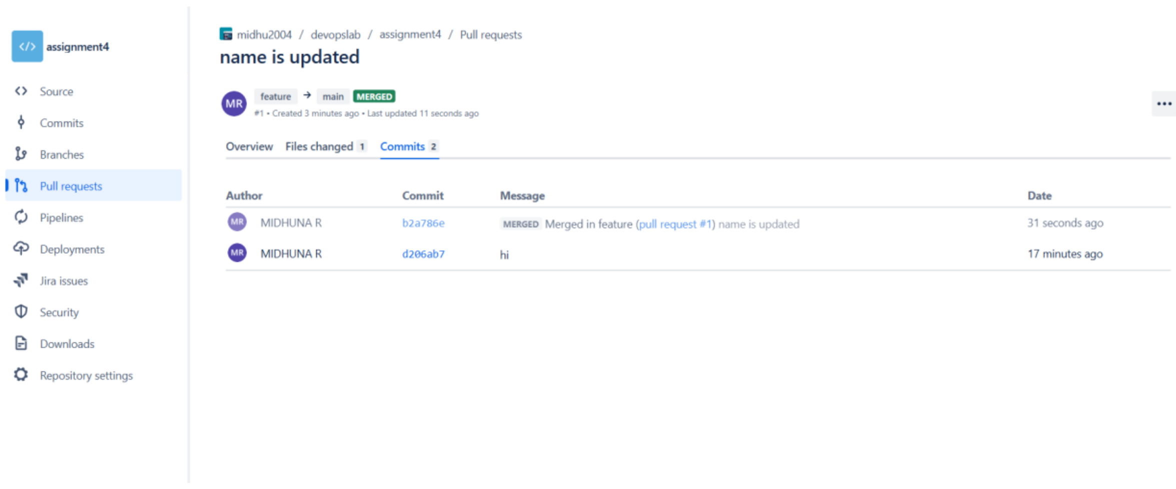
Activity All activity
What do you want to say?

TODAY
MIDHUNA R MERGED the pull request 1 minute ago

0 builds
It looks like you haven't configured a build tool yet. You can use Bitbucket Pipelines to build, test and deploy your code.
Your existing plan already includes build minutes.
Set up a pipeline

0 checks

.


assignment4
- Source
- Commits
- Branches
- Pull requests
- Pipelines
- Deployments
- Jira issues
- Security
- Downloads
- Repository settings

midhu2004 / devopslab / assignment4 / Pull requests

name is updated

MR feature → main MERGED
#1 • Created 3 minutes ago • Last updated 11 seconds ago

Overview Files changed 1 Commits 2

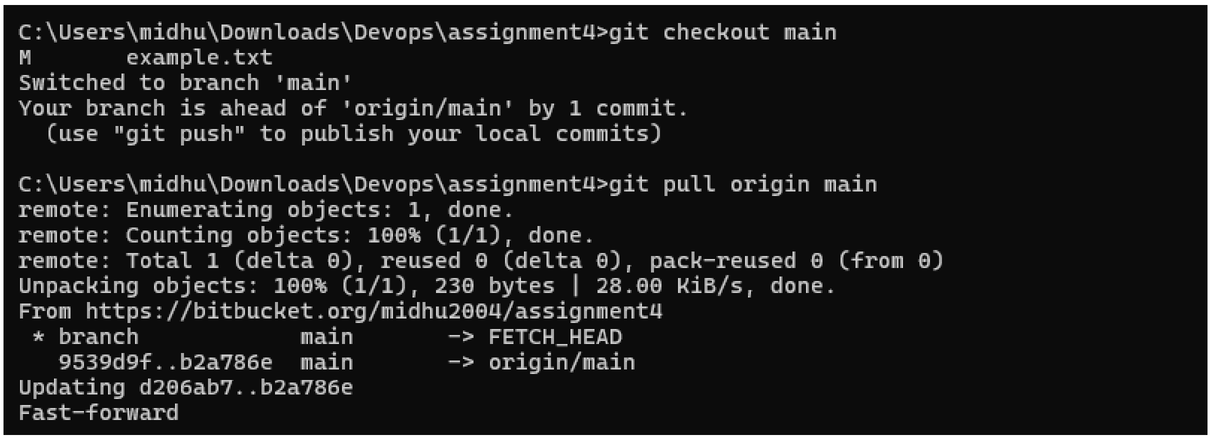
Author	Commit	Message	Date
MIDHUNA R	b2a786e	MERGED Merged in feature (pull request #1) name is updated	31 seconds ago
MIDHUNA R	d206ab7	hi	17 minutes ago

Step 6: Syncing Changes

- After the pull request is merged, update your local repository:

git checkout main

git pull origin main


```
C:\Users\midhu\Downloads\Devops\assignment4>git checkout main
M       example.txt
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

C:\Users\midhu\Downloads\Devops\assignment4>git pull origin main
remote: Enumerating objects: 1, done.
remote: Counting objects: 100% (1/1), done.
remote: Total 1 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (1/1), 230 bytes | 28.00 KiB/s, done.
From https://bitbucket.org/midhu2004/assignment4
 * branch            main       -> FETCH_HEAD
   9539d9f..b2a786e  main       -> origin/main
Updating d206ab7..b2a786e
Fast-forward
```

Conclusion:

This experiment provided you with practical experience in performing Bitbucket operations using Git commands. You learned how to create repositories, clone them to your local machine, make changes, create branches, push changes to Bitbucket, collaborate through pull requests, and synchronise changes with remote repositories. These skills are essential for effective collaboration and version control in software development projects using Bitbucket and Git.

Questions/Exercises:
Q1. What is Bitbucket, and How Does It Fit into the DevOps Landscape?

Bitbucket is a web-based Git repository hosting service that provides source code management (SCM) and collaboration tools for teams. It is part of the Atlassian suite and integrates with other Atlassian tools like Jira and Confluence, making it a versatile platform for development and project management.

Role in DevOps Landscape:

- **Version Control:** Supports Git repositories to manage codebase versions.
- **CI/CD Integration:** Integrates with Bitbucket Pipelines for automated builds, testing, and deployment.
- **Collaboration:** Facilitates team collaboration through pull requests and inline comments.
- **Scalability:** Offers cloud and self-hosted (Bitbucket Data Center) options.
- **Integration:** Seamlessly connects with other DevOps tools for issue tracking, monitoring, and deployment.

Q2. Explain the Concept of Branching in Bitbucket and Its Significance in Collaborative Development.

Branching in Bitbucket allows developers to work on separate versions of a codebase simultaneously.

Branching Concept:

- **Feature Branches:** Isolate work on new features from the main codebase.
- **Bug Fix Branches:** Address specific issues without disrupting ongoing development.
- **Release Branches:** Prepare and stabilize code for production releases.

Significance in Collaborative Development:

- **Parallel Development:** Multiple team members can work independently on different tasks.
- **Code Stability:** Keeps the main branch stable while allowing experimentation.
- **Flexibility:** Enables testing and review of individual branches before merging.
- **Traceability:** Tracks changes and contributions from each developer.

Q3. What Are Pull Requests in Bitbucket, and How Do They Facilitate Code Review and Collaboration?

A **pull request** in Bitbucket is a mechanism to propose, discuss, and merge changes from one branch into another.

How Pull Requests Facilitate Collaboration:

- **Code Review:** Allows team members to review and comment on code changes.
- **Collaboration:** Developers can discuss specific changes directly in the pull request.
- **Approval Workflow:** Requires reviewers to approve changes before merging, ensuring quality.
- **Integration Testing:** Triggers CI/CD pipelines to validate changes.
- **Documentation:** Tracks discussions and decisions for future reference.

Q4. How Can You Integrate Code Quality Analysis and Security Scanning Tools into Bitbucket's CI/CD Pipelines?

Bitbucket Pipelines allows integration of code quality and security tools to maintain high standards.

Steps to Integrate Tools:

1. **Choose a Tool:** Use tools like SonarQube, Checkmarx, or Snyk for quality and security analysis.
2. **Configure Pipeline Script:** Add commands in the bitbucket-pipelines.yml file to execute the tool during builds.

```
pipelines:
 default:
  - step:
     name: Code Quality and Security Check
     script:
      - sonar-scanner
      - snyk test
```

3. **Set Up Tool Integration:** Authenticate and configure tools as required (e.g., API keys).
4. **Analyze Reports:** Review the generated reports for vulnerabilities or code issues.
5. **Automate Actions:** Fail builds if critical issues are detected.

Benefits:

- Automated quality and security checks during development.
- Early detection of issues, reducing risk in production.

Q5. What Are Merge Strategies in Bitbucket, and How Do They Affect the Merging Process During Pull Requests?

A **merge strategy** determines how changes from a source branch are integrated into a target branch during a pull request.

Common Merge Strategies in Bitbucket:

1. **Merge Commit:** Creates a new commit for the merge, preserving the commit history.
    - **Use Case:** When tracking the history of branch merges is important.
2. **Squash:** Combines all commits from the source branch into a single commit.
    - **Use Case:** Simplifies history for small changes or single-feature branches.
3. **Rebase:** Reapplies commits from the source branch on top of the target branch.
    - **Use Case:** Maintains a linear commit history but requires caution to avoid conflicts.
4. **Fast-Forward:** Directly applies changes if the target branch has no new commits.
    - **Use Case:** Best for small, quick updates without conflicts.

Impact on Pull Requests:

- Determines commit history clarity and branch organization.
- Affects how conflicts are resolved during merging.
- Supports project workflows and team preferences for repository maintenance.