RQF LEVEL 3

SWDVC301

SOFTWARE DEVELOPMENT

# Version Control

TRAINER'S MANUAL

October, 2024

# VERSION CONTROL

2024

## AUTHOR'S NOTE PAGE (COPYRIGHT)

**© Rwanda TVET Board**

*Copies available from:*

- *HQs: Rwanda TVET Board-RTB*
- *Web: www.rtb.gov.rw*
- **KIGALI-RWANDA**

Original published version: October 2024

## ACKNOWLEDGEMENTS

The publisher would like to thank the following for their assistance in the elaboration of this training manual:
Rwanda TVET Board (RTB) extends its appreciation to all parties who contributed to the development of the trainer's and trainee's manuals for the TVET Certificate III in Software development, specifically for the module **"SWDVC301: Version Control."**

We extend our gratitude to KOICA Rwanda for its contribution to the development of these training manuals and for its ongoing support of the TVET system in Rwanda

We extend our gratitude to the TQUM Project for its financial and technical support in the development of these training manuals.

We would also like to acknowledge the valuable contributions of all TVET trainers and industry practitioners in the development of this training manual.

The management of Rwanda TVET Board extends its appreciation to both its staff and the staff of the TQUM Project for their efforts in coordinating these activities.

**This training manual was developed:**

Under Rwanda TVET Board (RTB) guiding policies and directives

Under Financial and Technical support of

## COORDINATION TEAM

RWAMASIRABO Aimable
MARIA Bernadette M. Ramos
MUTIJIMA Asher Emmanuel

## PRODUCTION TEAM

### Authoring and Review

MUKESHIMANA Anastase

BIZIMUNGU Damien

### Validation

NYABUHORO Elisabeth

HABIGENA Alexandre

KWIZERA Emmanuel

### Conception, Adaptation and Editorial works

HATEGEKIMANA Olivier

GANZA Jean Francois Regis

HARELIMANA Wilson

NZABIRINDA Aimable

DUKUZIMANA Therese

NIYONKURU Sylvestre

BIZIMANA Eric

### Formatting, Graphics, Illustrations, and infographics

YEONWOO Choe
SUA Lim
SAEM Lee
SOYEON Kim
WONYEONG Jeong
HAKIZAYEZU Adrien

### Financial and Technical support

KOICA through TQUM Project

# TABLE OF CONTENT

## ACRONYMS

**CBT:** Competency-Based Training
**CMD:** Command prompt
**CVS:** Concurrent Version System
**Git:** Global information tracker
**INIT:** Initialization
**Rm:** Remove
**RTB:** Rwanda TVET Board
**SVN:** Subversion
**TQUM Project:** TVET Quality Management Project
**TVET**: Technical and Vocational Education and Training
**URL:** Uniform resource locator

# INTRODUCTION

This trainer's manual includes all the methodologies required to effectively deliver the module titled **"Version Control."** Trainees enrolled in this module will engage in practical activities designed to develop and enhance their competencies.

The development of this training manual followed the Competency-Based Training and Assessment (CBT/A) approach, offering ample practical opportunities that mirror real-life situations.

The trainer's manual is organized into Learning Outcomes, which is broken down into indicative content that includes both theoretical and practical activities. It provides detailed information on the key competencies required for each learning outcome, along with the objectives to be achieved.

As a trainer, you will begin by asking questions related to the activities to encourage critical thinking and guide trainees toward real-world applications in the labor market. The manual also outlines essential information such as learning hours, didactic materials, and suggested methodologies.

This manual outlines the procedures and methodologies for guiding trainees through various activities as detailed in their respective trainee manuals. The activities included in this training manual are designed to offer students opportunities for both individual and group work. Upon completing all activities, you will assist trainees in conducting a formative assessment known as the end learning outcome assessment. Ensure that students review the key reading and the points to remember section.

## MODULE CODE AND TITLE: SWDVC301 VERSION CONTROL

|  |
|---|
| **Learning Outcome 1: Setup repository** |
| **Learning Outcome 2: Manipulate files** |
| **Learning Outcome 3: Ship codes** |

## Learning Outcome 1: Setup Repository

| Indicative contents |
|---|
| **1.1 Introduction to version control**<br><br>**1.2 Description of git**<br><br>**1.3 Use of GitHub repository** |

**Key Competencies for Learning Outcome 1: Setup repository**

| Knowledge | Skills | Attitudes |
|---|---|---|
| • Description of version control<br>• Description of Terminals used in version control<br>• Description of git<br>• Description of GitHub | • Using CMD terminal commands for directory management<br>• Installing git<br>• Preparing git environment<br>• Configuring .gitignore file<br>• Creating local repository<br>• Creating GitHub account<br>• Creating new remote repository<br>• Applying git commands related to repository | • Being Problem solver<br>• Have Team work spirit<br>• Have critical thinking<br>• Being self-motivation<br>• Being Adaptability<br>• Being Practical oriented |

**Duration: 20hrs**

**Learning outcome 1 objectives**:

By the end of the learning outcome, the trainees will be able to:

1. Describe clearly version control based on software development process
2. Use correctly terminals based on computer system available
3. Prepare properly git environment based on git command
4. Create properly git Repository based on the project requirements
5. Create correctly GitHub account based on project requirement
6. Create correctly remote repository based on software development standard
7. Set properly Remote URL in accordance with Git commands.

**Resources**

| Equipment | Tools | Materials |
|---|---|---|
| ● Computer | ● Git<br>● GitHub<br>● Text editor (vs code)<br>● Terminal (CMD, Gitbash). | ● Internet<br>● Electricity |

**Advance Preparation:**

Before delivering this learning outcome, you are recommended to:

- Avail git set up
- Avail files or project related to software development

## Indicative content 1.1: Introduction to version control.

**Duration: 5hrs**

**Theoretical Activity 1.1.1: Description of version control.**

**Notes to the trainer:**

- While delivering this content, a small group can be used for describing Version Control.

**Key steps:**

**While delivering this content pass through the following steps:**

**Step 1:** Introduce the activity and ask learners to answer the following questions:

1. What do you understand about Version Control?
2. What is Terminal in software development?
3. Describe the Different types of version control systems
4. Where version control is applied
5. What are the benefits of version control?

**Step 2**: Ask learner to write answers provided on flipchart/paper.
**Step 3**: Asks learners to discuss the provided answer and choose correct answers.
**Step 4**: Provides expert view and clarifies ideas.
**Step 5:** Address any questions or concerns.
**Step 6:** Ask trainees to read the key reading 1.1.1 in the trainee manual.

**Points to Remember**

- Version control system is about tracking files' changes, allows developer to collaborate
- There are primarily three types of version control systems: Local Version Control Systems, Centralized Version Control Systems, and Distributed Version Control Systems.
- Git and GitHub are Distributed Version Control Systems.
- There are most popular terminals commonly used with popular version control systems include: command prompt (CMD), PowerShell, Bash.

**Practical Activity 1.1.2: Using CMD commands**

**Notes to the trainer**

- While delivering this content, you are required to Check if computers have Command Prompt (CMD) as terminal.

**Key steps:**

**While delivering this activity, pass through the following steps:**

**Step 1:** Introduce the topic and ask trainees do the task described below:

As software developer, you are asked to go to the computer lab to use cmd commands to create a directory, change directory, rename directory, delete directory etc……

**Step 2**: Explain the task and provide clear work instruction.

**Step 3:** Ask trainees to read key reading 1.1.2

**Step 4**: Demonstrate how to use cmd commands. While demonstrating, explain the steps to follow.

**Step 5**: Asks learners to use cmd commands and monitor the procedures.

**Step 6:** Verify whether rd or rmdir Command, dir command, cd Command, cd.. Command in directory management are correctly used.

**Points to Remember**

- There are Three ways to open CMD
- Default directory is C:\Users\YourUsername
- Ensure the correct prefix is used for a respective CMD command
- Location of directory created and how to access it.

**Application of learning 1.1.**

As an IT you are responsible for managing directories in your own project workspace, you are asked with using the Command Prompt (CMD) to create directories called work1, and work2 within the "ProjectX" directory, change work2 by adding file called login.php, rename work1 to administration, finally delete work1 from projectX. This ensures efficient and precise management of your workspace's file system, optimizing organization and accessibility.

**Checklist**

| **Criteria** | **Indicator/Element to check** | **Observation** | |
|---|---|---|---|
| | | **yes** | **no** |
| Command prompt is run | Command prompt is properly run | | |
| Directories are created | work1 directory is created | | |
| | work2 directory is created | | |
| | Work1 and work2 are created within project directory | | |
| Directories are changed | The file called login.php is added to work2 | | |
| Directories are renamed | work1 is renamed to administration | | |
| Directories are deleted | work1 is deleted from projectX. | | |

## Indicative content 1.2: Description of Git

**Duration: 9hrs**

**Theoretical Activity 1.2.1: introduction of Git**

**Notes to the trainer:**

- While delivering this content, a small group can be used for introducing Git.
- The use of videos as didactic materials is required.

**Key steps:**

**While delivering this activity, pass through the following steps:**

**Step1**: Involve trainees in group formulation.

**Step2**: Introduce the activity and ask learners to respond to the following questions:

1. What do you understand about git?
2. Provide the description of:
   - Features of git
   - Git basic concepts
   - Git architecture and git workflow.

**Step3**: Ask trainees to present their findings to the whole class.

**Step4**: Provides expert view and clarifies ideas by using didactic materials.

**Step5:** Address any questions or concerns.

**Step6**: Ask trainee to read the key readings on activity 1.2.1

**Points to Remember**

- Git's architecture is designed around the concept of a distributed version control system, allowing users to work independently and collaborate seamlessly. There are the key components and concepts of Git's architecture including: Working Directory, Index (Staging Area), Local Repository, Remote Repository.
- Git is powerful distributed version control system, is built on several fundamental concepts that form the foundation of its functionality. Here are Git's basic concepts including: Repository, commit, branch, merge, pull, push.

## Practical Activity 1.2.2: Installing git set up

**Notes to the trainer**

- This activity should take place in a computer lab where trainees should install and configure the git setup.
- Avail computers
- Avail git setup

**Key steps:**

**While delivering this activity, pass through the following steps:**

**Step 1:** Introduce the topic and ask trainees do the task described below:

As a software developer, you are asked to go to the computer lab to install and configure git set up.

**Step 2:** Explain the task and provide clear work instruction.

**Step 3:** Ask trainees to read key reading 1.2.2.

**Step 4:** Demonstrate how to install the git set up. While demonstrating, explain the steps to follow.

**Step 5:** Ask learners to install the git set up and monitor the procedures.

**Step 6:** Verify whether the git installation and configuration are correctly performed.

**Points to Remember**

- **To install Git on Windows, follow these steps:**
  1. Browse to the official Git website: https://git-scm.com/downloads
  2. Click the download link for Windows and allow the download to complete.
  3. Browse to the download location
  4. Allow the app to make changes to your device
  5. Review the GNU General Public License
  6. specify the installation location.
  7. Leave the defaults component unless you have a specific need to change.
  8. create a start menu folder.
  9. Select a text editor you'd like to use with Git.
  10. choose a different name for your initial branch.
  11. change the PATH environment.
  12. select SSH client you want Git to use.
  13. switch to Windows Store certificates.
  14. converts line endings.
  15. Choose the terminal emulator you want to use.

16. The installer now asks what the git pull command should do.
17. Next you should choose which credential helper to use.
18. decide which extra option you would like to enable.
19. install experimental features.
20. Once the installation is complete, tick the boxes to view the Release Notes or Launch Git Bash.

- **Basic commands to Configure GIT**
  1. Git init command
  2. Git config command
  3. Git – version command

**Application of learning 1.2.**

XYZ Company, as part of your responsibilities, you need to showcase your expertise in Git for our software development project. This involves installing the current version of Git, verifying the installation, initializing a Git repository in the XYZ project folder, and configuring the .gitignore file to exclude dot env file paths for security purposes.

**Checklist**

| **Criteria** | **Indicator/element to check** | **Observation** | |
|---|---|---|---|
| | | **yes** | **no** |
| Git setup is installed | The current version of Git is installed | | |
| Installed git is verified | git --version command is applied | | |
| Git repository is initialized in the XYZ project folder | Git init command is used to initialize repository in given folder | | |
| The.gitignore file is configured | .gitignore file is created in XYZ project folder | | |

## Indicative content 1.3: Use of GitHub repository

**Duration: 6 hrs**

### Theoretical Activity 1.3.1: Description of GitHub

**Notes to the trainer:**

- While delivering this content, a small group can be used for describing GitHub.
- The use of videos as didactic materials is required.

**Key steps:**

**While delivering this activity, pass through the following steps:**

**Step 1:** Introduce the activity and request learners to respond to the following questions:

1. What do you understand about GitHub?
2. Provide an explanation of:
   - The features of GitHub
   - Benefits of GitHub
   - GitHub account
   - Remote repository
   - Git commands related to repository as used in GitHub.

**Step 2:** Asks any learner to write answers provided on flipchart/paper.

**Step 3:** Asks learners to discuss the provided answer and choose correct answers.

**Step 4:** Provides expert view and clarifies ideas by using didactic materials.

**Step 5:** Address any questions or concerns.

**Step 6:** Ask trainees to read the key reading 1.3.1 in the trainee manual.

**Points to Remember**

- GitHub, a widely used platform for software development and collaboration, offers a range of features that facilitate version control, project management, code sharing, and team collaboration. There are some key features of GitHub including: Collaboration, Integrated issue and bug tracking, Graphical representation of branches, Git repositories hosting, Project management, Team management.
- GitHub accounts are user profiles on the GitHub platform that provide individuals and organizations with access to various features and functionalities for managing and collaborating on software projects.

**Practical Activity 1.3.2: Using GitHub**

**Notes to the trainer**

- This activity should take place in a computer lab where trainees should create accounts on GitHub, create remote repositories in their account, apply the git config command and other commands related to repository.
- Avail computer connected to the internet.
- Avail Web Browser

**Key steps:**

**While delivering this activity, pass through the following steps:**

**Step 1:** Introduce the topic and ask learners do the task described below:

As a software developer, you are asked to go to the computer lab to use GitHub by creating account.

**Step 2:** Explain the task and provide clear work instructions.

**Step 3:** Ask trainees to read key reading 1.3.2.

**Step 4:** Demonstrate how to create account in GitHub. While demonstrating, explain the steps to follow.

**Step 5:** Asks learners to create GitHub account and monitor the procedures.

**Step 6:** Verify whether the Creation of GitHub account, Creation of repository in your account, git-config command, git commands related to repository used in GitHub are clearly performed.

**Points to Remember**

- To use GitHub effectively, follow these steps:

1. Create GitHub account
2. Create repository in your account
3. Perform git-config command
4. Perform git commands related to repository

**Application of learning 1.3.**

Alex, a software developer, is starting a new project to build a personal finance tracking application. Alex wants to use GitHub to manage the project's version control and collaborate with other developers. How Alex goes through the process to perform these tasks:

1. Set Up the Project Directory called project1
2. Initialize Git Repository

3. Create a GitHub Repository named products
4. Link Local Repository to GitHub
5. Add update1.php,  delete2.php, and  Login.php as Initial Files
6. Push update1.php and Login.php  to  the product repository
7. Collaborate with Other Developers

**Checklist**

| **Criteria** | **Indicators** | **observation** | |
|---|---|---|---|
| | | **Yes** | **No** |
| Project Directory is created | Directory called project1 is created | | |
| Git Repository is Initialized | Git init command is performed | | |
| GitHub Repository is Created | Product repository is created on GitHub | | |
| Local Repository is linked to GitHub | URL is available and work correctly | | |
| Initial Files are added | update1.php,  delete2.php, Login.php files are added | | |
| Files are Pushed  to GitHub | update1.php and Login.php  are pushed to products repository | | |
| Collabolation is made  with Other Developers | Added files are shared with developers | | |

## Learning outcome 1 end assessment

**Written assessment**

1. **Read the Following statement and answer by true if correct or false otherwise**

Version control systems (VCS), including distributed versions like Git, are essential tools in software development that enable tracking changes over time, facilitate collaboration, and maintain a comprehensive version history. Through concepts like branching and merging, developers can work on separate features or fixes independently and later integrate them into the main codebase. Key commands like git pull to fetch and merge changes, git status to view repository state, and git branch to manage branches, all contribute to efficient project management by providing snapshots of the project at specific points and allowing seamless collaboration.

a. Version control systems (VCS) track changes to files over time, enabling collaboration and version history.
b. Distributed version control systems (DVCS) allow each user to have a complete copy of the repository with its full history.
c. Branching in version control allows developers to work on separate features or fixes without interfering with the main codebase.
d. Merging in version control combines changes from one branch or fork into another, integrating separate lines of development.
e. Commits in version control systems are snapshots of the entire project at a specific point in time, including all tracked files.
f. git pull fetches changes from a remote repository and merges them into the local repository.
g. git status displays information about the current state of the repository, such as modified files and branch status.
h. git branch is used to create, list, delete, or manipulate branches in a Git repository.

2. Match Git commands with its corresponding description

| Command | Description |
|---|---|
| Git Commit | 1. Manages connections to remote repositories. |
| Git Clone | 2. Copies a repository from a remote source to your local machine. |
| Git Push | 3. Fetches and merges changes from a remote repository to your local repository. |
| Git Pull | 4.Removes files from the working directory and stages the removal for the next commit |
| Git Remote | 4.Records changes to the repository |
| Git Status | 5. Uploads local repository content to a remote repository. |
| Git Config | 6. Sets configuration options for Git on your local machine. |
| Git rm | 7.Shows the current state of the repository, including tracked/untracked files and changes |

3. read the following sentences and Fill the gap with the missing word.

1. The command git __________ is used to create a new branch in Git.
2. A Git __________ is a pointer to a specific commit in the repository's history.
3. git checkout __________ is used to switch between branches in Git.
4. To list all branches in a Git repository, you can use git branch __________.
5. A Git __________ is a copy of a repository that lives on your computer instead of on a website's server.
6. git merge __________ is used to integrate changes from one branch into another.
7. git remote __________ is used to manage connections to remote repositories.
8. git push origin __________ is used to push the changes of the current branch to a remote repository.
9. A centralized version control system offers ………………………….a way to collaborate using a central server.
10. Distributed version control system allows ………………… management branching and merging.

4. Respond by **True** or **False** from the followings:

a) Git bash is one among version control system that exist
b) Git init is used to clone remote repository
c) Git status is used to initialize an empty repository
d) In git you can view untracked files by using git version command

**Answers**

1.**Answer by True or False**

a. True
b. True
c. True
d. True
e. True
f. True
g. True
h. True

**2.Match Git commands with its corresponding description**

**Git Commit:** 5. Records changes to the repository
**Git Clone: 2**. Copies a repository from a remote source to your local machine
**Git Push: 6**. Uploads local repository content to a remote repository
**Git Pull: 3**. Fetches and merges changes from a remote repository to your local repository
**Git Remote:** 1. Manages connections to remote repositories
**Git Status: 8**. Shows the current state of the repository, including tracked/untracked files and changes
**Git Config: 7.** Sets configuration options for Git on your local machine
**Git rm**: 4. Removes files from the working directory and stages the removal for the next commit

**3**. **Fill the gap in the following sentences.**

a) The command git **branch** is used to create a new branch in Git.
b) A Git **commit** is a pointer to a specific commit in the repository's history.
c) git checkout **branch-name** is used to switch between branches in Git.
d) To list all branches in a Git repository, you can use git branch **--list**.
e) A Git **clone** is a copy of a repository that lives on your computer instead of on a website's server.
f) git merge **branch-name** is used to integrate changes from one branch into another.
g) git remote **add** is used to manage connections to remote repositories.
h) git push origin **branch-name** is used to push the changes of the current branch to a remote repository.
i) A centralized version control system offers **software development teams** a way to collaborate using a central server.
j) Distributed version control system allows **automatic** management branching and merging**.**

4. **Respond by True or False from the followings:**

**Answer:**

**a) Git bash is one among version control systems that exist. False.**

- Git Bash is not a version control system; it is a command-line terminal emulator and shell that provides an environment for running Git commands. Git itself is the version control system.

**b) Git init is used to clone remote repositories. False.**

- git init is used to initialize a new Git repository in a local directory. It is not used to clone a remote repository. Cloning a remote repository is typically done with git clone**.**

**c) Git status is used to initialize an empty repository. False.**

- git status is used to display the current state of a Git repository, including information about untracked files, changes to be committed, and more. It is not used to initialize an empty repository. To initialize a Git repository, you would use git init.

**d) In git, you can view untracked files by using the git version command. False.**

- The git version command is used to display the installed Git version. To view untracked files in a Git repository, you can use the git status command, not git version.

**Practical assessment**

Mugabe XY holds the position of Senior Developer at Innovate Company Ltd, situated in Ruhango District. He has delegated a project to four developers, tasking them with designing a web application featuring various forms: login, student registration, course registration, and book registration, all using HTML. However, due to geographical constraints and other commitments, the developers found it challenging to collaborate effectively on the project. Consequently, the Senior Developer opted to assign tasks to each developer individually and remotely, recommending them to work autonomously on their designated tasks by referring to this repository structure.

```
├── login.html
├── student_registration.html
├── course_registration.html
├── book_registration.html
├── README.md
└── .gitignore
```

As one of the developers at the company, you have been assigned the following task:

Create a remote repository on GitHub using your full name as the repository name.

Clone the repository to your local computer.

Within the repository, create the required forms mentioned above for subsequent commits.

**Checklist**

| **Criteria** | **Indicator/element to check** | **observation** | |
|---|---|---|---|
| | | **yes** | **no** |
| Git environment is prepared | Git setup is installed | | |
| | Git is configured | | |
| GitHub account is created | GitHub account is available | | |
| The required forms are created inside that repository | Login, Student Registration, Course Registration, and Book Registration forms are created | | |
| Repository structure is respected | Repository structure is followed | | |
| Remote repository is created | Remote repository is created with your full name | | |
| created repository is cloned | The repository is cloned to your local computer | | |

**Further information to the trainer**

JavaTpoint. (n.d.). Git Version Control System. Retrieved from https://www.javatpoint.com/git-version-control-system

JavaTpoint. (n.d.). IntelliJ IDEA Version Control. Retrieved from https://www.javatpoint.com/intellij-idea-version-control

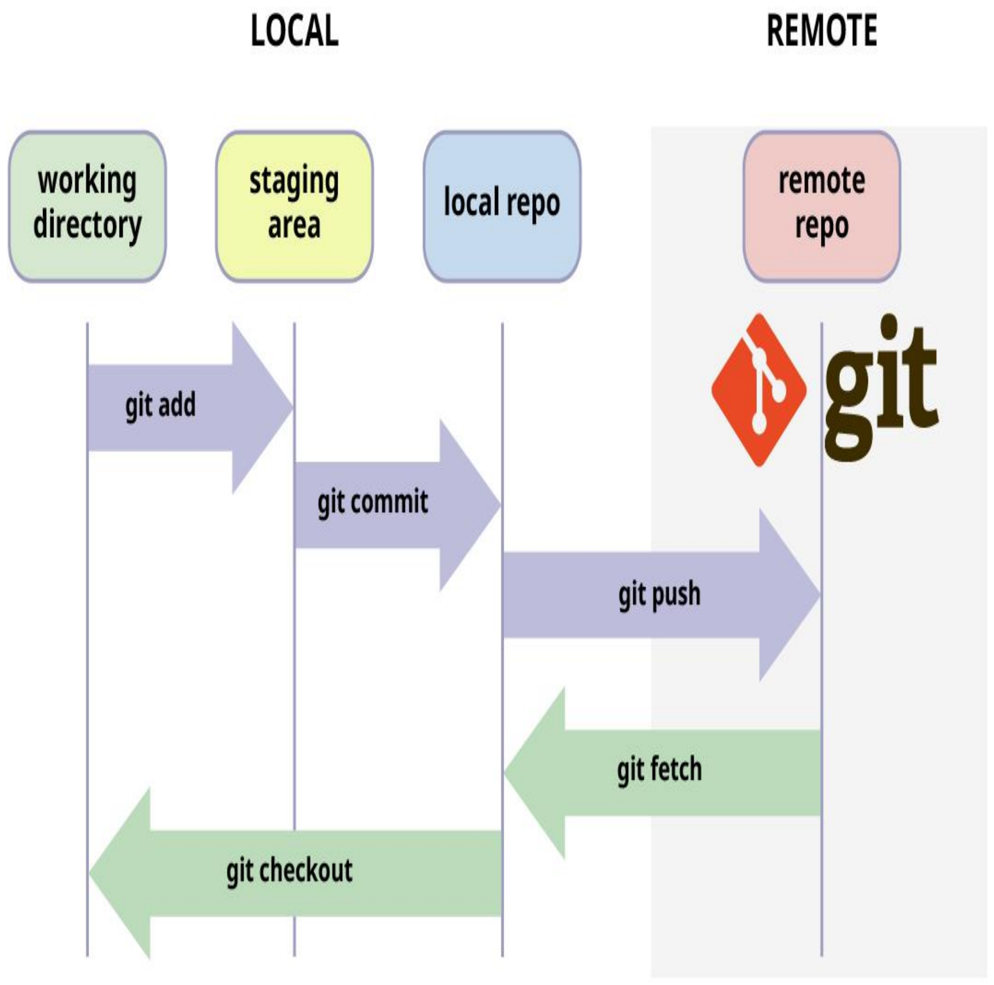
Red Hat Developers. (2023, August 2). Beginner's guide to Git version control. Retrieved from https://developers.redhat.com/articles/2023/08/02/beginners-guide-git-version-control#

JavaTpoint. (n.d.). Git. Retrieved from https://www.javatpoint.com/git

JavaTpoint. (n.d.). How to Install Git on Windows. Retrieved from https://www.javatpoint.com/how-to-install-git-on-windows

TutorialsPoint. (n.d.). Git Environment. Retrieved from https://www.tutorialspoint.com/git/git_environment.htm

JavaTpoint. (n.d.). GitHub. Retrieved from https://www.javatpoint.com/github


LOCAL
REMOTE
working directory
staging area
local repo
remote repo
git
git add
git commit
git push
git fetch
git checkout

| **Indicative contents** |
|---|
| **2.1: Adding file change to git staging area** |
| **2.2: Commit file changes to git local repository and manage branches** |

**Key Competencies for Learning Outcome 2: Manipulate files**

| Knowledge | Skills | Attitudes |
|---|---|---|
| • Description of git status command<br>• Description of git commands operations<br>• Description of commit message<br>• Introduction of branches operations | • Applying git status commands<br>• Performing Operation on git add command<br>• Using staging area<br>• Performing File management commands<br>• Applying git commit command<br>• Applying Operations on git branches<br>• Adding file change to git staging area | • Being Adaptability<br>• Being Practical oriented<br>• Have Communication skills<br>• Have Creativity<br>• Have critical thinking<br>• Being Problem solver<br>• Have Team work |

| |
|---|
| **Duration: 20 hrs** |
| **Learning outcome 2 objectives**:<br>By the end of the learning outcome, the trainees will be able to:<br>1. Describe clearly git status command based on git command.<br>2. Describe properly git command operations based on the project requirements.<br>3. Introduce clearly operations on branches based on project requirements.<br>4. Add correctly file change to git staging area based on operations.<br>5. Apply correctly git commit command based on project content.<br>6. Perform properly file management commands based on project requirements.<br>7. Apply clearly operations on branches based on project standard. |
| **Resources** |

| **Equipment** | **Tools** | **Materials** |
|---|---|---|
| ● Computer | ● Git<br>● GitHub<br>● Text editor (vs code)<br>● Terminal (CMD, Gitbash). | ● Internet<br>● Electricity |

**Advance Preparation:**

Before delivering this learning outcome, you are recommended to:

- Avail of computers connected to the internet
- Prepared git environment
- Avail website files like html files or php files

## Indicative content 2.1: Add file change to git staging area

**Duration: 10hrs**

**Theoretical Activity 2.1.1: Description of staging area**

**Notes to the trainer:**

- While delivering this content, a small group can be used for describing staging area.
- The use of videos as didactic materials is required.

**Key steps:**

**While delivering this activity, pass through the following steps:**

**Step 1**: Involves trainees to form groups

**Step 2**: Introduce the activity and request learners to respond to the following questions:

1. Describe the operations of these commands:
    - Git status command.
    - git add command
    - git reset command
    - Rm command

**Step 3:** Ask trainees to present their findings to the whole class.

**Step 4:** Provides expert view and clarifies ideas by using didactic materials.

**Step5:** Ask trainees to read key readings in activity 2.1.1

**Points to Remember**

- Git status is a useful command in Git that provides information about the current state of the repository and helps users track changes to files. There are operations related to `git status` including: View new untracked files, View modified files, View deleted files.
- The `git add` operation in Git is a fundamental command used to stage changes in a repository, preparing them to be included in the next commit. There are operations related to the `git add` command including: Stage all files, stage a file, Stage a folder.

## Practical Activity 2.1.2: Adding file change to git staging area

**Notes to the trainer**

- This activity should take place in a computer lab where trainees should perform git commands related to add file change to staging area like (git status, add, reset and rm) used in git.
- While delivering this content, you are required to:
  - Avail computer connected to the internet.
  - Avail computer installed with git
  - Avail html files/ php files and other related files
  - Avail text editor

**Key steps:**

**While delivering this activity, pass through the following steps:**

**Step 1**: Introduce the topic and ask trainees to perform the following tasks:

As software developer, you are requested to go to the computer lab to apply add file change to git staging area.

**Step 2**: Provide instruction that will be followed.

**Step 3:** Demonstrate how to add file change to git staging area, while demonstrating, explain the steps to follow.

**Step 4:** Ask trainees to read key reading 2.1.2.

**Step 5:** Asks learners to add file change to git staging area and monitor the procedures.

**Step 6:** Verify whether git status, add, reset and rm commands used in git are clearly performed.

**Points to Remember**

**To add file changes to the Git staging area, follow these steps:**

1. Check the Status
2. Add Changes
3. Stage All Changes
4. Review Staged Changes
5. Commit Staged Changes

**Application of learning 2.1.**

You are a software developer working on a coding project, who utilize Git as version control. While adding a new feature to the project, you need to create a file called "feature.html" and made modifications to an existing file called "main.html". To be sure that the previous tasks are performed well you can also review the changes before committing them. Using Git, check the status of your project with "git status" and identify the "feature.html" file as untracked, and "main.html" as modified. Use “git diff main.html" to view the differences in "main.html". During this process, you discover an unnecessary file and deleted it.

**Checklist**

| **Criteria** | **Indicators** | **Observation** | |
|---|---|---|---|
| | | **Yes** | **No** |
| New features are added to a project | Feature.html file is created | | |
| | Main.html is modified | | |
| Changes are committed | Git commit command is performed | | |
| | Git status command is performed | | |
| The git diff command is performed | the differences in "main.html" is viewed | | |

## Indicative content 2.2: Commit File changes to git local repository and manage branche.

**Duration: 10 hrs**

**Theoretical Activity 2.2.1: Introduction of commit file change to git local repository**

**Notes to the trainer:**

- While delivering this content, a small group can be used for introducing commit file change to git local repository.
- The use of videos as didactic materials is required.

**Key steps:**

**While delivering this activity, pass through the following steps:**

**Step1**: Involve trainees in group formulation

**Step 2**: Introduce the activity and request learners to respond to the following questions:

1. What do you understand about commit message?
2. What are the best practices for creating a commit message, operations related to git log?
3. Can you explain the operations involved in the `git commit` command in Git?

**Step 3:** Monitor the tasks

**Step 4:** Provides expert view and clarifies ideas by using didactic materials.

**Step 5**: Ask trainees to read the Key readings 2.2.1 in their manuals.

**Points to Remember**

- When using the `git commit` command in Git, several operations are applied to create a new commit in the repository. There are operations involved in the `git commit` command: Staging, Creating the commit, recording the commit message, committing to the local repository, advancing the branch pointer.

**Practical Activity 2.2.2: Committing File changes to git local repository**

**Notes to the trainer**

- This activity should take place in a computer lab where trainees should perform git commit and git log command operation to commit file changes to git local repository and perform different branch operations used in git.
- While delivering this content, you are required to:
    - Avail computer connected to the internet.
    - Avail computer installed with git
    - Avail files to add in staging area
    - Avail text editor

**Key steps:**

**While delivering this activity, pass through the following steps:**

**Step 1:** Introduce the topic and ask learners do the task described below:

As a Software Developer, you are asked to go to the computer lab to commit file changes to git local repository and manage branches.

**Step 2:** Explain the task and provide clear work instruction.

**Step 3:** Ask trainees to read key reading 2.2.2.

**Step 4:** Demonstrate how to commit file changes to git local repository and manage branches.

**Step 5:** Ask learners to commit file changes to git local repository and manage branches and monitor the procedures.

**Step 6:** Verify whether the git commit, git log command operation to commit file changes to git local repository and branch operations used in git are correctly performed.

**Points to Remember**

**Here are the steps to commit file changes to a Git local repository:**

1.Check Status
2. Stage Change
3. Verify Staging
4. Commit Changes
5. Verify Commit

**Application of learning 2.2.**

Three students participating in project development as team, here are tasks to perform:, Alice takes charge of creating a new branch called **feature-xyz** to develop a new feature. She wants to manage her changes and review the commit history to ensure that her work is well-documented. Meanwhile, Bob, another team member, is working on a separate feature on the feature-abc branch. Both they want to keep track of available branches and switch between them. Alice periodically merges the latest changes from the main branch, supervised by Sarah, the project lead, to ensure her branch remains up to date. Together, Alice, Bob, and Sarah effectively collaborate using Git's powerful features to manage their workflow, track progress, and maintain code integrity.

**Checklist**

| **Criteria** | **Indicators** | **observation** | |
|---|---|---|---|
| | | **Yes** | **No** |
| Branche are created | Feature-xyz branch is created | | |
| Branches are switched one to another | Switching between feature-abc and feature-XYZ is made | | |
| The last changes are merged | Alice merges the latest changes from the main branch | | |
| Collaboration is apply | Alice, Bob, and Sarah collaborate using Git’s powerful features | | |

## Learning outcome 2 end assessment

**Written assessment**

1.Match the Git branch operations with their corresponding commands:

| **Operation** | **Command** |
|---|---|
| Create branch | a. git branch <branch-name> |
| List branch | b. git branch |
| Delete local branch | c. git branch -d <branch-name> |
| Delete remote branch | d. git push origin --delete <branch-name> |
| Switch branch | e. git checkout <branch-name> |
| Rename branch | f. git branch -m <old-branch-name> <new-branch-name> |

2. Complete the sentence:

I. Before staging a file, you must check all..........files and..........files.
II. To create a new branch in Git, you use the command git ___________ <branch-name>.
III. To list all branches in a Git repository, you use the command git ___________.
IV. To delete a local branch, you use the command git ___________ <branch-name>.
V. To delete a remote branch, you use the command git ___________ origin --delete <branch-name>.
VI. To switch to a different branch, you use the command git ___________ <branch-name>.
VII. To rename a branch, you use the command git ___________ <old-branch-name> <new-branch-name>.

3. How do I add files to a commit?

a. $ git stage
b. $ git commit
c. $ git add
d. $ git reset

4. How to save the current state of your code in git?

a. By validating the modifications staged with $ git commit
b. By adding all the changes and staging them with $ git stage
c. By adding all the changes and organizing them with $ git add
d. By creating a new commit with $ git init

5. **Read the Following statement and answer by true if correct or false otherwise**

The git add command is crucial for staging changes, whether they involve new files or modifications to existing ones, preparing them for the next commit. Once staged, the git commit command is used to permanently save these changes to the local repository, typically requiring a commit message for clarity. Additionally, the git commit command can be employed in the process of merging branches within Git, integrating changes from different lines of development.

a. The git add command is used to stage changes for the next commit.
b. The git add command can be used to stage both new files and modifications to existing files
c. The git commit command is used to permanently save changes to the local repository.
d. The git commit command requires a commit message to be provided.
e. The git commit command can be used to merge branches in Git.

**Answers**

1. Match the Git branch operations with their corresponding commands:

| **Operation** | **Command** |
|---|---|
| Create branch | a. git branch <branch-name> |
| List branch | b. git branch |
| Delete local branch | c. git branch -d <branch-name> |
| Delete remote branch | d. git push origin --delete <branch-name> |
| Switch branch | e. git checkout <branch-name> |
| Rename branch | f. git branch -m <old-branch-name> <new-branch-name> |

**2. Complete the sentence:**

i. Before staging a file, you must check all **untracked** files and **modified** files.
ii. To create a new branch in Git, you use the command **git branch <branch-name>.**
iii. To list all branches in a Git repository, you use the command **git branch.**
iv. To delete a local branch, you use the command **git branch -d <branch-name>.**
**v.** To delete a remote branch, you use the command **git push origin --delete <branch-name>.**
**vi.** To switch to a different branch, you use the command **git checkout <branch-name>.**
**vii.** To rename a branch, you use the command **git branch -m <old-branch-name> <new-branch-name>.**

**3.** How do I add files to a commit?

✓ $ git add

4. How to save the current state of your code in git?

✓ By validating the modifications staged with $ git commit

**5.** Answer by **true** or **false**.

a. The git add command is used to stage changes for the next commit. **True**

b. The git add command can be used to stage both new files and modifications to existing files. **True**

c. The git commit command is used to permanently save changes to the local repository. **True**

d. The git commit command requires a commit message to be provided. **True**

e. The git commit command can be used to merge branches in Git. **False** (Merging branches is typically done using git merge.)

**Practical assessment**

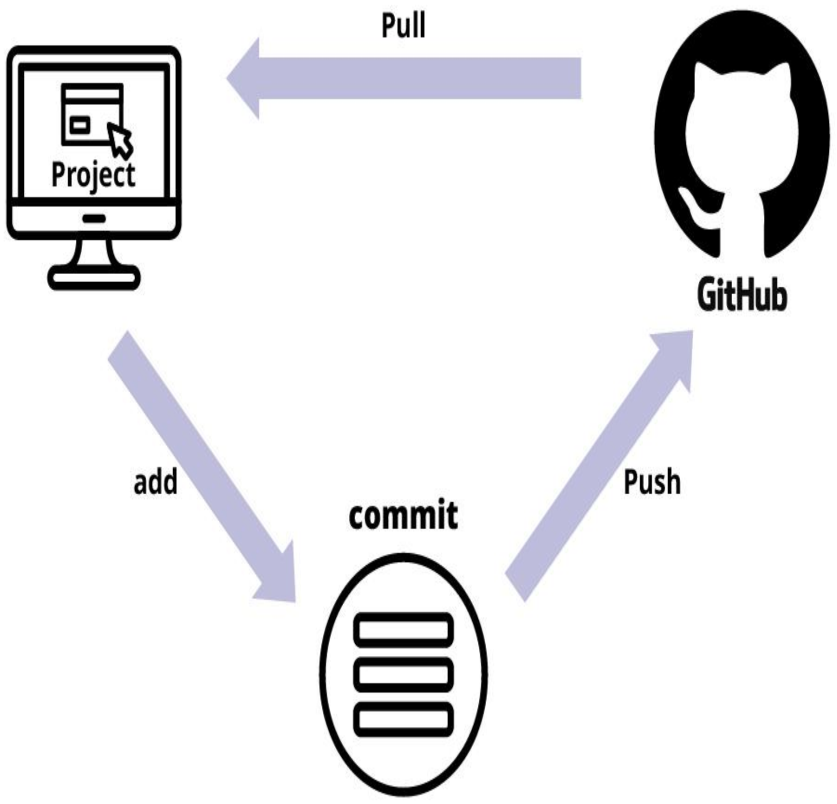
As the project's owner, Sarah wants to tweak a few features and has assigned her coworkers assignments to do so. She begins by creating a new Git repository called maths and a branch called "feature_branch." By adding a new file with the name new_file.txt, Alex makes modifications to the project files. Emily verifies that the repository is functioning properly, adds new_file.txt to the staging area, and then commits the modifications with the message "Add new_file.txt." David makes a switch to the main branch, merges the updates from "feature_branch," and clears up any conflicts. Finally, Sarah, the repository's owner, removes the "feature_branch" following a successful merge. They handle the project's branches, use the staging area properly, and commit file changes to the local Git repository.

**Checklist**

| **Criteria** | **Indicators** | **Observation** | |
|---|---|---|---|
| | | **Yes** | **No** |
| Git repository is created | Maths repository is created | | |
| Branch is created | Feature_branch branch is created | | |
| Files are added | New_file.txt is added | | |
| All modifications are saved | Commit command is performed | | |
| Commits message is created. | "Add new_file.txt." commit message is created | | |
| Merging is applied | Updates are merged from "feature_branch," | | |
| conflicts are handled | Conflicts from merging are cleaned | | |
| Project's branches are handled and save file changes | "feature_branch," is removed and commint the change | | |

**Further information to the trainer**

LinuxHint. (n.d.). Git: List of new, modified, and deleted files. Retrieved from https://linuxhint.com/git-list-of-new-modified-deleted-files/
Noble Desktop. (n.d.). How to stage and commit files in Git. Retrieved from https://www.nobledesktop.com/learn/git/stage-commit-files
JavaTpoint. (n.d.). Git Reset. Retrieved from https://www.javatpoint.com/git-reset
JavaTpoint. (n.d.). Git rm. Retrieved from https://www.javatpoint.com/git-rm
CareerFoundry. (n.d.). Git commit command explained. Retrieved from https://careerfoundry.com/en/blog/web-development/git-commit-command/
TutorialsPoint. (n.d.). Git Managing Branches. Retrieved from https://www.tutorialspoint.com/git/git_managing_branches.htm


Pull
Project
GitHub
add
commit
Push

| Indicative contents |
|---|
| **3.1: Fetch file from GitHub repository**<br><br>**3.2: Push files to remote branch**<br><br>**3.3: Merge branches on remote repository** |

**Key Competencies for Learning Outcome 3: Ship codes**

| Knowledge | Skills | Attitudes |
|---|---|---|
| • Description of pull and fetch commands operations<br>• Description of pull request<br>• Explanation of Tags used on git push command and operations<br>• Description of operations on git rebase command<br>• Description of operation on git merge. | • Fetching file from GitHub repository<br>• Pulling files to GitHub repository<br>• Pushing files to remote branch<br>• Creating pull request<br>• Merging branches on remote repository | • Being Practical oriented<br>• Have Communication Skills<br>• Have critical thinking<br>• Have Team work spirit<br>• Being Problem solver |

|  |  |  |
|---|---|---|
| **Duration: 20 hrs** | | |
| **Learning outcome 3 objectives**:<br>By the end of the learning outcome, the trainees will be able to:<br>1. Describe correctly pull, fetch, push, and git rebase commands based on git project requirement<br>2. Fetch correctly files in different operations based on git project structure<br>3. Push properly files to remote branch based on committed files<br>4. Merge effectively branches on remote repository based on pull request created. | | |
| **Resources** | | |
| **Equipment** | **Tools** | **Materials** |
| ● Computer | ● Git<br>● GitHub<br>● Text editor (vs code)<br>● Terminal (CMD, Gitbash). | ● Internet<br>● Electricity |
| **Advance Preparation:**<br>Before delivering this learning outcome, you are recommended to:<br>• Avail computers connected to the internet<br>• prepared of git environment<br>• Avail GitHub account<br>• Avail command line/terminal in his/ her computer<br>• Avail remote repository link or a Git remote configured.<br>• Avail website files like html files or php files | | |

## Indicative Content 3.1: Fetch file from GitHub repository

**Duration: 7hrs**

**Theoretical Activity 3.1.1: introduction of Fetching file from GitHub**

**Notes to the trainer:**

- While delivering this content, a small group can be used for introducing fetch file from GitHub.

**Key steps:**

**While delivering this activity, pass through the following steps:**

**Step1**: Involves trainees in group formulation

**Step2**: Introduce the activity and request learners to respond to the following questions:

1. What do you understand about the following term?
   - Fetch
   - Pull
2. Can you explain the operations involved in the `git fetch` command in Git?
3. Can you explain the operations involved in the `git pull` command?

**Step3**: Monitor the task

**Step4:** Ask trainees to present their findings to the whole class.

**Step4**: Provides expert view and clarifies ideas by using didactic materials.

**Step5:** Ask trainees to read the key reading 3.1.1. in trainee manual

**Points to Remember**

**The git fetch command** is a powerful tool in Git that allows you to retrieve changes from a remote repository and update your local repository accordingly.

**The git pull command** is used to fetch and merge changes from a remote repository into your local branch.

The process of delivering or deploying code changes from a development environment to a production environment or making them available to users involves several key steps including Fetch file, Push files with corresponding operations

**Practical Activity 3.1.2: Fetching file from GitHub repository**

**Notes to the trainer**

- This activity should take place in a computer lab where trainees should perform git fetch command and pull commands in different operations like Fetch the remote repository ,Fetch the specific branch ,Fetch all the branch simultaneously, Synchronize the local repository ,Default git pull , Git pull remote branch , Git force pull , Git pull origin master used in git.
- While delivering this content, you are required to:
  - Avail computer connected to the internet.
  - Avail computer installed with git
  - Avail branches create in GitHub
  - Avail text editor

**Key steps:**

**While delivering this activity, pass through the following steps:**

**Step 1:** Introduce the topic and ask trainees do the task described below:

As a software developer, you are asked to go to the computer lab to fetch file from GitHub repository.

**Step 2:** Explain the task and provide clear work instruction.

**Step 3:** Ask trainees to read key reading 3.1.2.

**Step 4:** Demonstrate how to fetch file from GitHub repository, while demonstrating, explain the steps to follow.

**Step 5:** Asks learners to fetch file from GitHub repository and monitor the procedures.

**Step 6:** Verify whether git fetch command and pull commands in different operations like Fetch the remote repository ,Fetch the specific branch ,Fetch all the branch simultaneously, Synchronize the local repository ,Default git pull , Git pull remote branch , Git force pull , Git pull origin master used in git are clearly performed.

**Points to Remember**

To fetch a file from a GitHub repository, you typically need to clone the repository to your local machine. Here are the steps:

1.Find the Repository

2.Get the Repository URL

3.Open Terminal/Command Prompt

4.Clone the Repository

5.Navigate into the Cloned Repository
6. Fetch the File

**Application of learning 3.1.**

The owner of the GBY project, which uses Git for version control, needs to improve it by adding new features and content in order to make management, collaboration, and maintenance easier. Different tasks are given to a team of developers to complete. As with getting the most recent updates from a cloud-based document collaboration platform, Alice is responsible with obtaining the remote repository by making subscription to umurunga channels, Bob is in charge of fetching a tasks branch. Claire's job is to synchronize the local repository called performance across all branches at once, which is similar to syncing a music streaming service across several devices. David executes the standard git pull action, which is comparable to an email client automatically downloading new emails. Emma "pulls" a particular remote branch which is tasks5 and comparable to downloading specific files or directories from cloud storage. In a manner similar to overwriting local files on a file synchronization service, Frank manages the git pull --force procedure. Last but not least, Grace updates a piece of software to the most recent stable version by pulling changes from the "master" branch.

**Checklist**

| **Criteria** | **Indicators** | **Observation** | |
|---|---|---|---|
| | | **Yes** | **No** |
| Subscription to other channels is made | Alice is subscribed to umurunga channels | | |
| Branches are fetched | Tasks branch is fetched | | |
| The local repositories are synchronized | performance repository synchronized across all branches at once | | |
| Git pull is performed | Standard git pull action is performed | | |
| | an email of client is automatically downloaded | | |
| Files and directories are comparable to what downloaded from cloud storage | Tasks5 is pulled and comparable to downloaded files | | |
| the git pull --force procedure is performed | Pull –force command is applied | | |
| | pulling changes from the "master" branch is made | | |

## Indicative content 3.2: Push files to remote branch

**Duration: 6hrs**

**Theoretical Activity 3.2.1: introduction to files pushing to remote branch**

**Notes to the trainer:**

- While delivering this content, a small group can be used for introducing file pushing to remote branch.
- The use of picture as didactic materials to show where files are moved from and other parts they pass through.

**Key steps:**

**While delivering this activity, pass through the following steps:**

**Step 1:** Introduce the activity and request learners to respond to the following questions:

1. What do you understand about the term push?
2. Provide a description of the tags used in the `git push` command.

**Step 2:** Asks any learner to write answers provided on flipchart/paper.

**Step 3:** Asks learners to discuss the provided answer and choose correct answers.

**Step 4:** Provides expert view and clarifies ideas by using didactic materials.

**Step 5:** Address any questions or concerns.

**Step 6**: Ask trainees to read the key reading 3.2.1 in the trainee manual.

**Points to Remember**

Tags in Git are used to mark specific points in history, such as releases or important commits. There are the tags used with the git push command like :<repository>,<refspec>,--all,--prune,--mirror,--dry-run,--tags.

**Practical Activity 3.2.2: Pushing files to remote branch**

**Notes to the trainer**

- Trainer engages trainees to Apply git push on origin master, git push force, and git push verbose finally delete a remote branch individually.
- While delivering this content, you are required to:
  - Avail computer connected to the internet.
  - Avail computer installed with git
  - Avail website files like html/php and other related files
  - Avail text editor

**Key steps:**

**While delivering this activity, pass through the following steps:**

**Step 1:** Introduce the topic and ask learners do the task described below:

As a software developer, you are asked to go to the computer lab to push files to remote branch.

**Step 2:** Explain the task and provide clear work instruction.

**Step3:** Asks trainees to read key readings in activity 3.2.2

**Step 4:** Demonstrate how to push files to remote branch. While demonstrating, explain the steps to follow.

**Step 5:** Asks learners to push files to remote branch and monitor the procedures.

**Step 6**: Verify whether push file on origin master, Git push force, Git push verbose, delete remote branch operation is clearly performed. are correctly performed.

**Points to Remember**

**To push files to a remote branch in Git, you typically follow these steps:**

1.Add and Commit the Changes

2. Check Remote Repository Status
3. Fetch and Pull Changes
4. Push Changes to Remote Branch
5. Verify Changes on Remote Repository

**Application of learning 3.2.**

As a member of the team project called "TechApp," you are assigned to implement a new feature named "UserAnalytics" and merge it into the main codebase by performing these process:

a. Push your local changes for the "UserAnalytics" feature to the origin master branch to make them accessible to your team.
b. Resolve conflicts arising from recent updates by forcefully updating the remote repository. Use the verbose option to get detailed information about the push operation.
c. Utilize the verbose option during the push process to track detailed progress and troubleshoot any issues.
d. After resolving conflicts and successfully pushing changes, merge your "UserAnalytics" feature into the main codebase.
e. Delete the remote branch for "UserAnalytics" after the feature has been successfully merged and is no longer needed.

**Checklist**

| Criteria | Indicators | Observation | |
|---|---|---|---|
| | | **Yes** | **No** |
| Merge into the main codebase is done | UserAnalytics is merged into code base | | |
| Pushing your local changes to the "origin master is performed | Team members can access master branch with UserAnalytic changes | | |
| Forcefully update the remote repository is done | detailed information about the push operation is available | | |
| The verbose option is applied | Track progress and troubleshoot issues are done | | |
| The remote branches are deleted | branch that is no longer needed for "UserAnalytics are deleted | | |

## Indicative content 3.3: Merge branches on remote repository

**Duration: 7hrs**

**Theoretical Activity 3.3.1: Description of merge branches on remote repository**

**Notes to the trainer:**

- While delivering this content, a small group can be used for describing merge branches on remote repository.

**Key steps:**

**While delivering this activity, pass through the following steps:**

**Step1**: Engage trainees in group formulation.

**Step2**: Introduce the activity and request learners to respond to the following questions:

I. Can you discuss the operation on git rebase command, pull requests, operation on git merge

**Step3**: Monitor the task

**Step4**: Ask trainees to present their findings to the whole class.

**Step5:** Provides expert view and clarifies ideas by using didactic materials.

**step6:** Ask trainees to read the key reading 3.3.1. in trainee manual

**Points to Remember**

The "git rebase" command in Git allows you to reapply a series of commits on top of another base commit, effectively moving the entire branch to begin from the tip of another branch. Key operations involved in using "git rebase" include starting a rebase, Resolve Conflicts, Continue Rebase, Abort Rebase, Skip Commits, Skip Commits.

**Practical Activity 3.3.2: Merging branches on remote repository**

**Notes to the trainer**

- Engages trainees to use git rebase command, create pull request and apply git merge operations (merge the specified commit to current active branch, merge commits into the master branch, git merge branch).
- While delivering this content, you are required to:
  - Avail computer connected to the internet.

- Avail computer installed with git
- Avail GitHub account with created branches
- Avail text editor

**Key steps:**

**While delivering this activity, pass through the following steps:**

**Step 1:** Introduce the topic and ask trainees do the task described below:

As web developer, you are asked to go to the computer lab to merge branches on remote repository.

**Step 2**: Explain the task and provide the work instructions

**Step 3:** Ask trainees to read key reading 3.3.2.

**Step 4**: Demonstrate how to merge branches on remote repository while demonstrating, explain the steps to follow.

**Step 5**: Asks learners to merge branches on remote repository and monitor the procedures.

**Step 6:** Verify whether git rebase command, create pull request and git merge operations used in git are clearly performed.

**Points to Remember**

**To merge branches on a remote repository in Git, you typically follow these steps:**

1.Fetch Remote Changes
2.Checkout the Target Branch
3.Merge the Branch
4.Push Changes to Remote Repository

**Application of learning 3.3.**

**Application of learning 3.3**

**MyApp** is a web application project targeted at enhancing the user authentication procedure that is being worked on by a team of developers. The following actions were carried out by the appropriate team members. The developer, John, used the "git rebase" command to merge the most recent modifications from the main branch into his feature branch, ensuring that his improvements were based on the most recent codebase. Sarah, the team leader, issued a pull request to merge John's branch into the main branch, allowing for easier collaboration and review. This enabled the team to handle the issue as a whole and confirm that the proposed solution was in line with the project's goals and criteria. Finally, after reading and approving "this is merging exercise" pull request, David as project manager use the merge procedure, thereby Implementing the upgraded user authentication feature properly and contributing to the overall solution of improving the application's security and user experience.

**Checklist**

| Criteria | Indicators | Observation | |
|---|---|---|---|
| | | Yes | No |
| Git rebase" command is performed | modifications from the main branch into feature branch is done | | |
| Pull request is created | John's branch is merged into the main branch | | |
| The upgraded user authentication is Implemented | Application's security and user experience is improved | | |
| A given pull request is created | this is merging exercise pull request is created | | |

## Learning outcome 3 end assessment

**Written assessment**

1. Which of these Git client commands creates a copy of the repository and a working directory in the client's workspace. (Choose one.)
   a) update
   b) checkout
   c) clone
   d) import
   e) None of the above
2. True or False? In Git, if you want to make your local repository reflect changes that have been made in a remote (tracked) repository, you should run the pull command.
   a) True
   b) False
3. Now, imagine that you have a local repository, but other team members have pushed changes into the remote repository. What Git operation would you use to download those changes into your working copy?
   a) checkout
   b) commit
   c) export
   d) pull
   e) update
   f) a, b, and c

4. fill-in-the-blank space related to fetching, pulling, and pushing files in Git
   a) To fetch changes from a remote repository without merging them into your local branch, you use the command `git ___________`.
   b) To fetch changes from a remote repository and merge them into your local branch, you use the command `git ___________`.
   c) To upload your local repository changes to a remote repository, you use the command `git ___________ origin <branch-name>`.
   d) To see the status of your local repository, including which branch you are on and any changes that are staged for commit, you use the command `git ___________`.
   e) To add all changes in your working directory to the staging area, you use the command `git ___________ .`.
   f) To commit your staged changes with a message, you use the command `git ___________ -m "Your commit message"`.
   g) To create a connection to a new remote repository, you use the command `git remote ___________ <name> <URL>`.
   h) To view the details of your remote repository connections, you use the command `git remote ___________`

**5.Match the operations with corresponding commands**

| **Operation** | **Command** |
|---|---|
| Fetch changes from remote repository | a. Git fetch |
| Fetch and merge changes from remote | b. Git pull |
| Push changes to remote repository | c. Git push origin <branch-name> |
| See the status of the local repository | d. Git status |
| Add all changes to the staging area | e. Git add |
| Commit staged changed with a message | f. Git commit – m "your commit message" |
| Create a connection to a remote repo | g. Git remote add <name><URL> |

**Answers to the assessment:**

1. Which of these Git client commands creates a copy of the repository and a working directory in the client's workspace? (Choose one.)

   **c) clone**
2. True or False? In Git, if you want to make your local repository reflect changes that have been made in a remote (tracked) repository, you should run the pull command.

   **a) True**
3. Now, imagine that you have a local repository, but other team members have pushed changes into the remote repository. What Git operation would you use to download those changes into your working copy?

   **d) pull**

**4. Fill-in-the-blank space related to fetching, pulling, and pushing files in Git:**

a) To fetch changes from a remote repository without merging them into your local branch, you use the command **git fetch**.

b) To fetch changes from a remote repository and merge them into your local branch, you use the command **git pull**.

**c)** To upload your local repository changes to a remote repository, you use the command **git push origin <branch-name>.**

d) To see the status of your local repository, including which branch you are on and any changes that are staged for commit, you use the command **git status**.

**e)** To add all changes in your working directory to the staging area, you use the command **git add ..**

**f)** To commit your staged changes with a message, you use the command **git commit -m "Your commit message".**

**g)** To create a connection to a new remote repository, you use the command **git remote add <name> <URL>.**

**h)** To view the details of your remote repository connections, you use the command **git remote -v.**

**5. Match the operations with corresponding commands:**

| **Operation** | **Command** |
|---|---|
| Create branch | a. git branch <branch-name> |
| List branch | b. git branch |
| Delete local branch | c. git branch -d <branch-name> |
| Delete remote branch | d. git push origin --delete <branch-name> |
| Switch branch | e. git checkout <branch-name> |
| Rename branch | f. git branch -m <old-branch-name> <new-branch-name> |

**Practical assessment**

ABC Software Solutions is a leading software development company, Alex and Sarah are two experienced developers who need to collaborate on a project aimed at resolving a critical bug in a mission-critical software system for a major client. The bug has caused disruptions in the client's operations, and immediate attention is required to rectify the issue. Alex takes on the responsibility of simulating updates from the client and making changes to a separate branch to identify the root cause of the bug. Additionally, Alex merges the client's changes into project branch to test potential fixes and improve the overall stability of the system. Throughout the process, Alex actively contributes by making modifications, committing them, and pushing their branch to the remote repository, ensuring that the bug fixes are well-documented and can be easily shared with the client. Sarah, an integral team member, communicates her additional changes to Alex, addressing specific edge cases and proposing enhancements to optimize the software system's performance. After pushing her branch to the remote repository, Alex ensures synchronization by fetching Sarah's updated branch and merging it into their own, incorporating the valuable contributions made by Sarah. ABC Software Solutions demonstrates their commitment to providing high-quality software solutions and meeting their client's critical needs by leveraging efficient collaboration, version control, and code management practices to swiftly address and resolve complex problems in their client's software systems.

**Checklist**

| **Criteria** | **Indicators** | **Observation** | |
|---|---|---|---|
| | | **Yes** | **No** |
| Files are fetched | The changes are committed to the branch. | | |
| | updates are fetched from the remote repository | | |
| The overall stability of the system is improved | Alex make modifications, commit and push project to the remote repository | | |
| Files are pushed | Emily's' modifications are pushed to the remote repository | | |
| Branches are merged | The main branch is merged to Emily branch. | | |

## Further information to the trainer

JavaTpoint. (n.d.). Git Pull. Retrieved from https://www.javatpoint.com/git-pull
JavaTpoint. (n.d.). Git Push. Retrieved from https://www.javatpoint.com/git-push
Varonis. (n.d.). Git Branching: A guide to working with branches. Retrieved from https://www.varonis.com/blog/git-branching
TutorialsPoint. (n.d.). Git Managing Branches. Retrieved from https://www.tutorialspoint.com/git/git_managing_branches.htm
Atlassian. (n.d.). Git Merge: Everything you need to know about merge in Git. Retrieved from https://www.atlassian.com/git/tutorials/using-branches/git-merge

RTB | RWANDA TVET BOARD

October, 2024