Datamites
Global Institute for Data Science

VERSION CONTROL WITH GIT

© 2023 DataMites™. All Rights Reserved | www.datamites.com

PURPOSE OF VERSION CONTROL

Datamites
Global Institute for Data Science

- **History Tracking:** Maintain a record of changes made to a project, including who made them, when, and what changed i.e., Traceability.Collaborate with others and helps in reviewing the code.
- **Collaboration:** Enable multiple developers to work on a project simultaneously, coordinating efforts and avoiding conflicts.
- **Code Backup:** Provide secure backups of project files to prevent data loss.
- **Feature Development:** Allow for the development of new features without impacting the main project.
- **Code Review**: Facilitate peer code review to ensure code quality and identify issues early.

© 2023 DataMites™. All Rights Reserved | www.datamites.com

GIT INTRODUCTION

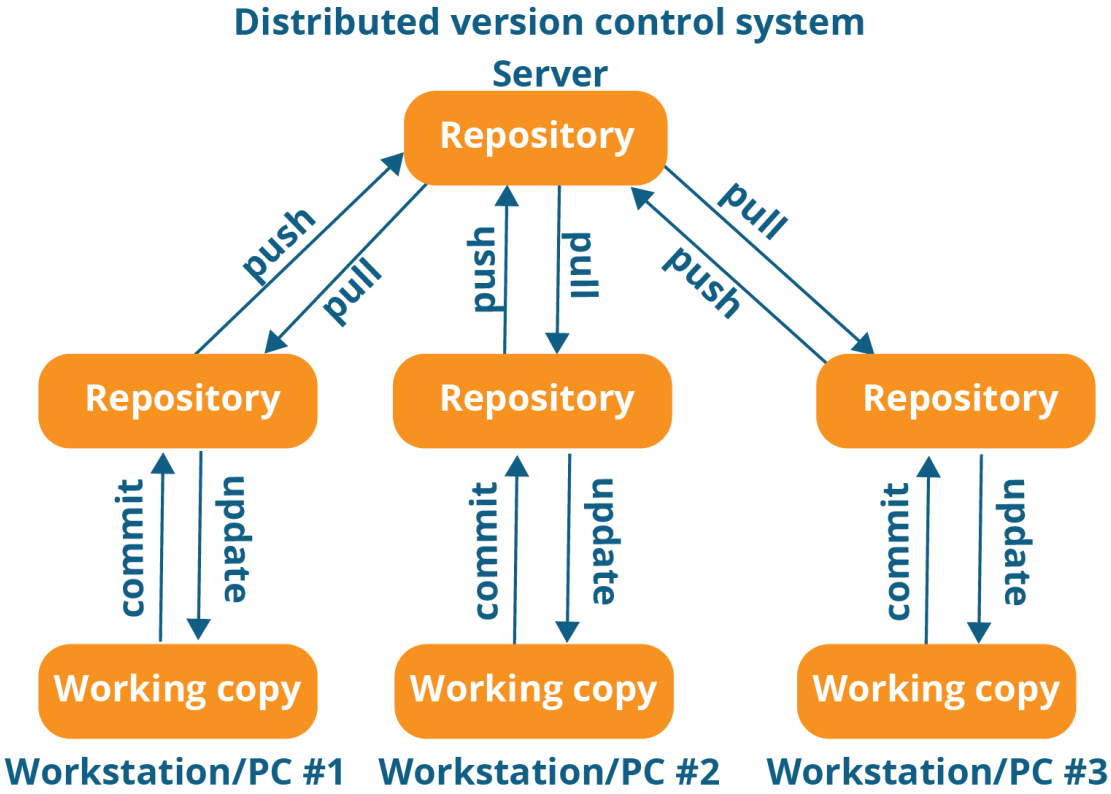
Datamites
Global Institute for Data Science

- Git is a source code management software.
- Git is a command-line tool i.e., A software which is maintained by Linux.
- Git is Software but Github, Gitlab, Bitbucket, etc…are all Services.
- It is **Distributed** version control system, meaning that each user has a copy of the entire project history on their local machine.

© 2023 DataMites™. All Rights Reserved | www.datamites.com

DISTRIBUTED VERSION CONTROL

Datamites
Global Institute for Data Science



© 2023 DataMites™. All Rights Reserved | www.datamites.com

TERMINOLOGIES IN GIT

Datamites
Global Institute for Data Science

- **Repository:** A Git repository is a place where Git stores all the versions of your project's files and directories.
- **Working Directory:** Where you see files physically and can do modification.
- **Staging Area (Index):** The staging area is an intermediate area where you can selectively choose which changes to include in your next commit.
- **Commit:** A commit is a snapshot of your repository at a specific point in time. It records the changes made to the local repository.
- **Commit Id/Version-Id:** Reference to identify each change.
- **Snapshot:** Represent some data of particular time. It is always incremental i.e., It stores the changes only, not the entire data.
- **Tag:** Tag assign a meaningful name with a specific version in the Repository. It's used to mark significant points in a project's history, such as releases or milestones.

© 2023 DataMites™. All Rights Reserved | www.datamites.com

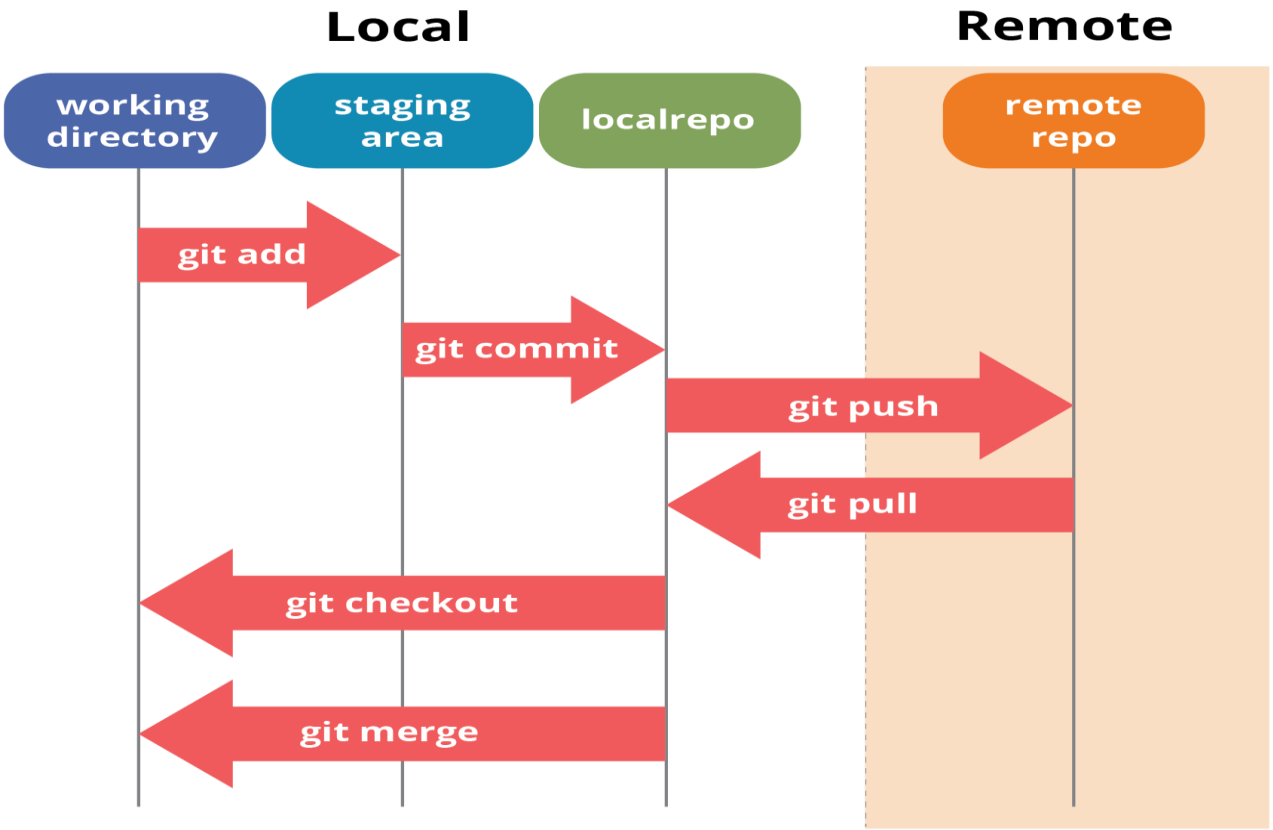
TERMINOLOGIES IN GIT

Datamites
Global Institute for Data Science

- **Pull:** Pulling is the process of retrieving changes from a remote repository to a Local Machine.
- **Push:** Pushing is the process of sending your local changes to a remote or Central (global)repository.
- **Server:** It stores all Repositories.
- **Branch:** A branch in Git is a parallel version of a codebase that enables multiple developers to work on different aspects of a project simultaneously.
- **Master (Main) Branch:** The master (or main) branch is the default branch in Git, and it typically represents the latest stable version of the project.
- **Fork**: Forking a repository creates a copy of it under your account, allowing you to make changes independently of the original repository.
- **HEAD**: It is a symbolic reference to the currently checked out branch or commit. It represents your current working state.

© 2023 DataMites™. All Rights Reserved | www.datamites.com

GIT WORKFLOW AND ARCHITECTURE



© 2023 DataMites™. All Rights Reserved | www.datamites.com

IMPORTANT COMMANDS IN GIT

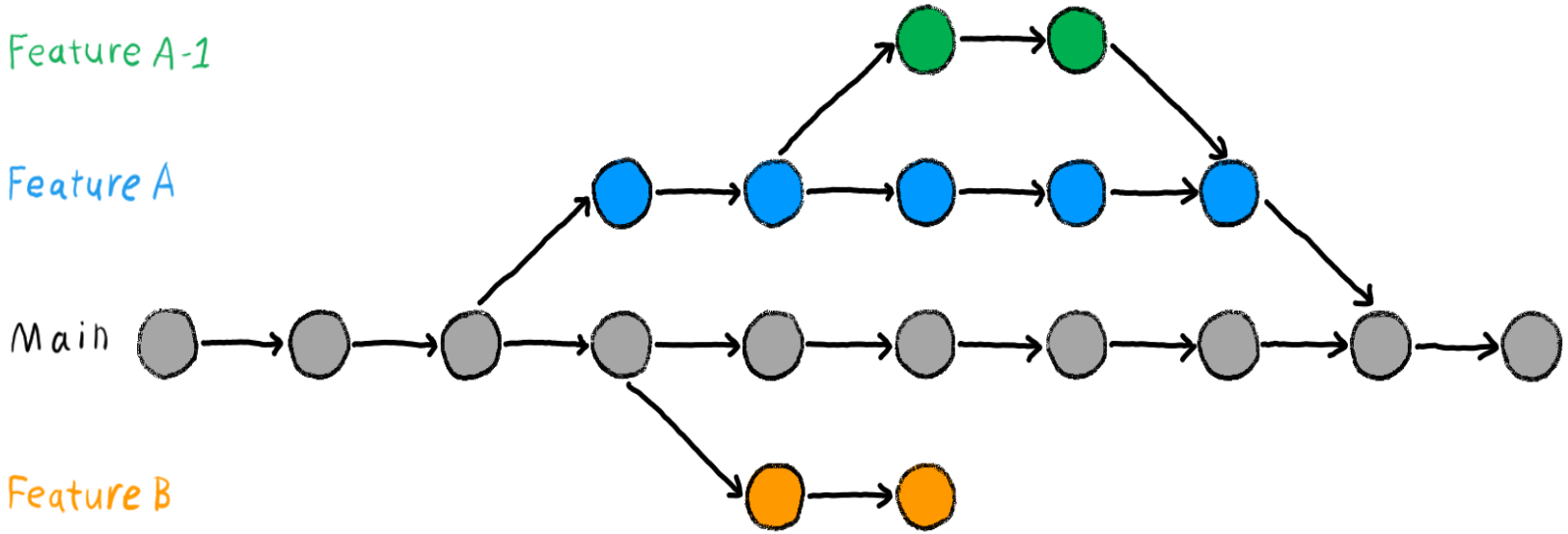
Datamites
Global Institute for Data Science

- **Git Config:-** This command sets the author name and email address to be used with your commits.
- **Git init:-** This command is used to initialize a local repository.
- **Git add:-** This command is used to add one or more files to staging (Index) area.
- **Git commit -m:-** This command create snapshots of the file permanently in the version history with a message.
- **Git Status:-** The status command is used to display the state of the working directory and the staging area.
- **Git push:-** This command is used to upload local repository content to a remote repository.
- **Git pull:-** This command is used to receive data from GitHub. It fetches and merges changes on the remote server to your working directory.

© 2023 DataMites™. All Rights Reserved | www.datamites.com

GIT BRANCHING, TAGGING AND MERGING

Datamites
Global Institute for Data Science



© 2023 DataMites™. All Rights Reserved | www.datamites.com

GIT BRANCHING, TAGGING AND MERGING

Datamites
Global Institute for Data Science

- **Branching** in Git allows you to create separate lines of development for features, bug fixes, or experiments, while keeping your main codebase (usually in the master branch) stable.

  - To Create a new branch: git branch new-feature
  - To Switch to the new branch: git checkout new-feature

- Git **tagging** is a way to mark specific points in your Git history, often used to identify releases or important milestones in the project.

- **Merging** in Git is the process of combining changes from one branch into another. This is commonly used to integrate feature branches into the main codebase i.e., to master branch.

© 2023 DataMites™. All Rights Reserved | www.datamites.com

ADVANTAGES OF USING GITHUB

Datamites
Global Institute for Data Science

- **Free and Open Source:-** Its free and Open Source Service.
- **Fast and Small:-** As most of the operations are performed locally, therefore it is fast.
- **Security:-** git uses a common Cryptographic hash function called Secure hash function(SHA1), So it's very Secure.
- **No-need of Powerful Hardware:-** With Git, we can easily work with large codebases, collaborate with other developers, and manage complex branching and merging workflows, all without needing a powerful machine.

© 2023 DataMites™. All Rights Reserved | www.datamites.com