Benson P Sabu

501641

IT Sem 8

**DevOps Exp 2**

**Aim: -** Version Control System using Git

**Theory: -**

Version control, also known as source control, is the practice of tracking and managing changes to software code. Version control systems are software tools that help software teams manage changes to source code over time. As development environments have accelerated, version control systems help software teams work faster and smarter. They are especially useful for DevOps teams since they help them to reduce development time and increase successful deployments.

Version control software keeps track of every modification to the code in a special kind of database. If a mistake is made, developers can turn back the clock and compare earlier versions of the code to help fix the mistake while minimizing disruption to all team members. A repository is a VCS term which describes when VCS is tracking a filesystem. In the scope of individual source code files, a VCS will track additions, deletions, modifications of the lines of text within that file.

Git is a software for tracking changes in any set of files, usually used for coordinating work among programmers collaboratively developing source code during software development. Its goals include speed, data integrity, and support for distributed, non-linear workflows (thousands of parallel branches running on different systems). As with most other distributed version control systems, and unlike most client–server systems, every Git directory on every computer is a full-fledged repository with complete history and full version-tracking abilities, independent of network access or a central server.

**GitHub:**

GitHub, Inc. is a provider of Internet hosting for software development and version control using Git. It offers the distributed version control and source code management (SCM) functionality of Git, plus its own features. It provides access control and several collaboration features such as bug tracking, feature requests, task management, continuous integration and wikis for every project. GitHub offers its basic services free of charge. Its more advanced professional and enterprise services are commercial. The GitHub flow is a lightweight, branch-based workflow built around core Git commands used by teams around the globe—including ours. The flow has six steps, each with distinct benefits when implemented:

***1. Create a branch​:*** Topic branches created from the canonical deployment branch (usually main) allow teams to contribute to many parallel efforts. Short-lived topic branches, in particular, keep teams focused and result in quick ships.

***2. Add commits:​*** Snapshots of development efforts within a branch create safe, revertible points in the project’s history.

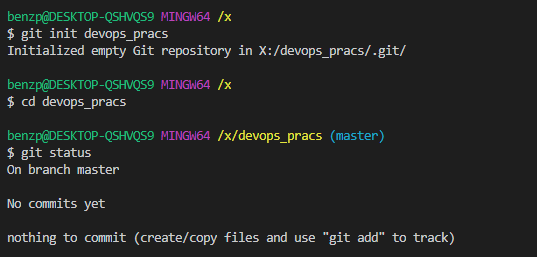
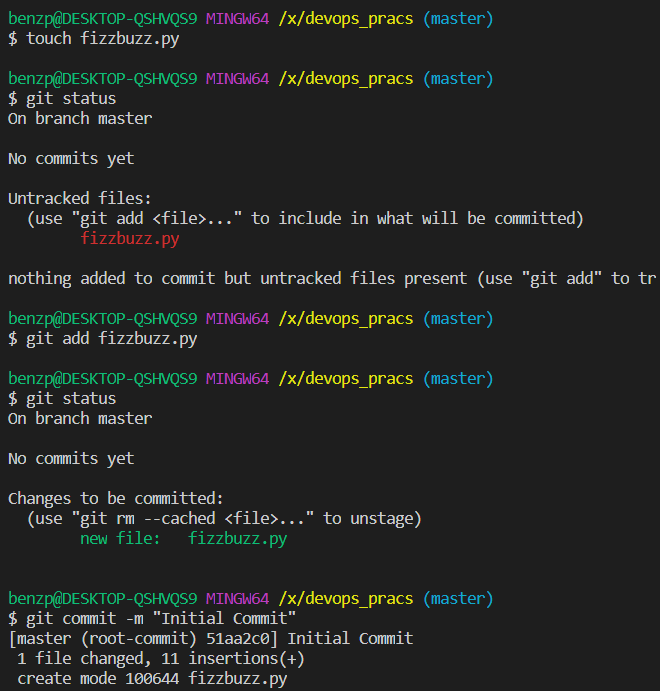
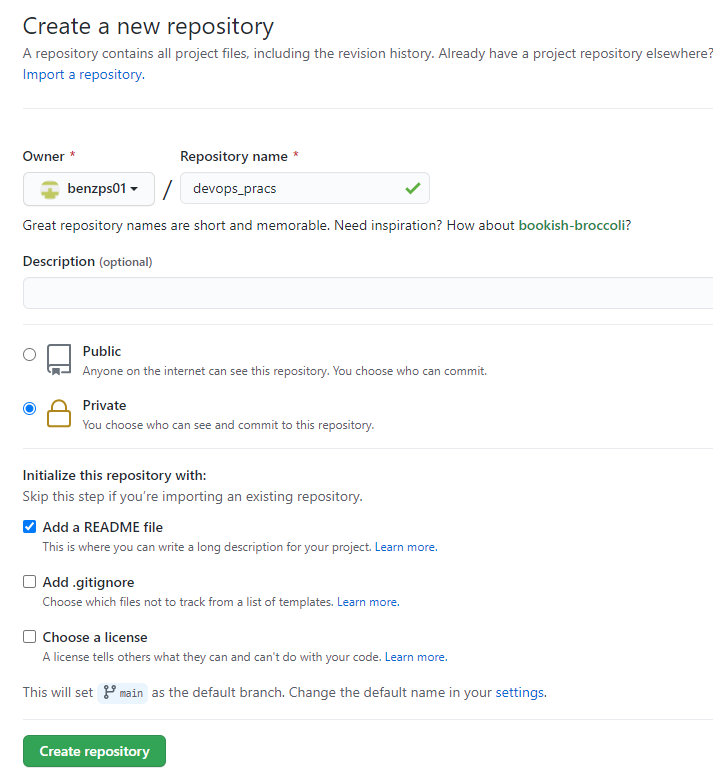
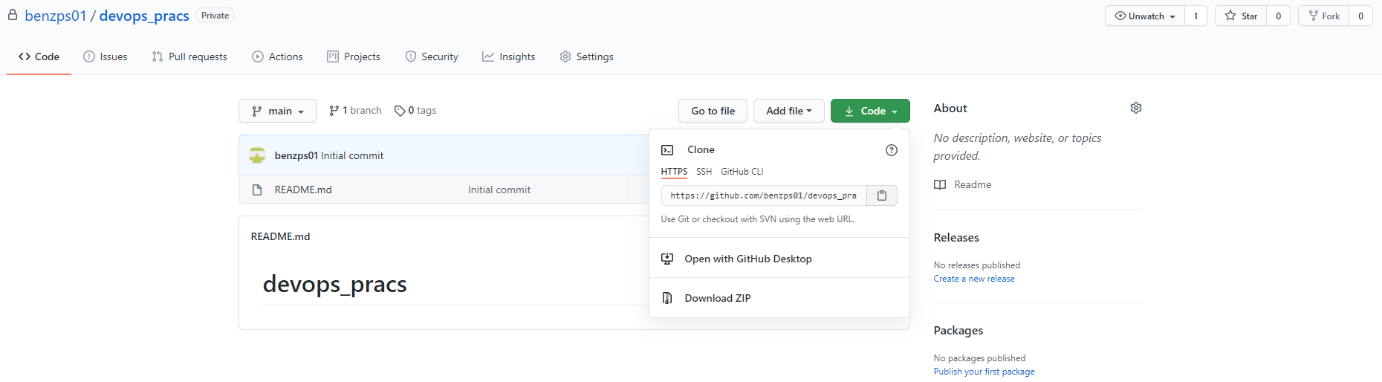
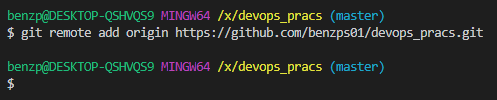
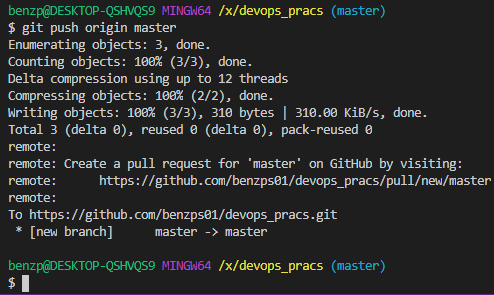
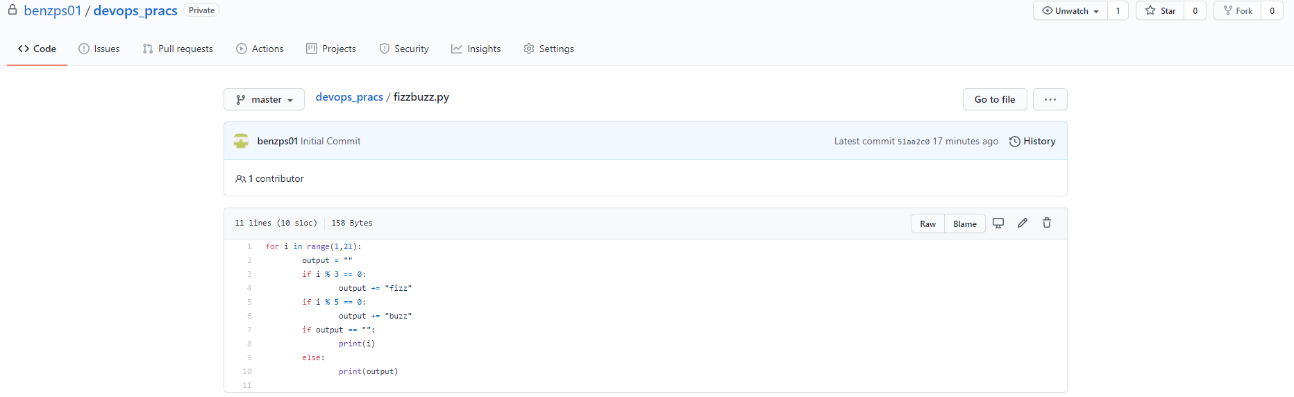
***3. Open a pull request:​*** Pull requests publicize a project’s ongoing efforts and set the tone for a transparent development process.

***4. Discuss and review code:​*** Teams participate in code reviews by commenting, testing, and reviewing open pull requests. Code review is at the core of an open and participatory culture.

***5. Merge:​*** Upon clicking merge, GitHub automatically performs the equivalent of a local ‘git merge’ operation. GitHub also keeps the entire branch development history on the merged pull request.

***6. Deploy:​*** Teams can choose the best release cycles or incorporate continuous integration tools and operate with the assurance that code on the deployment branch has gone through a robust workflow.

**Creating a repository and adding files:**

1. Creating a repository using “git init” command.
2. Adding files to the repository and committing them locally
3. Create a remote repository on GitHub.
4. Copy the web URL once the repository is created.
5. Sync the local repository with remote repository.
6. Push the changes to remote repository
7. Check the remote repository on GitHub for the changes made.

Commands used:

* Init
* Status
* Add
* Commit
* Remote
* Push

**Conclusion: -** Hence, Version Control System was studied and successfully implemented using Git.