***Experiment 1***

**Aim** **:** Installation of Git and creating Git Repository.

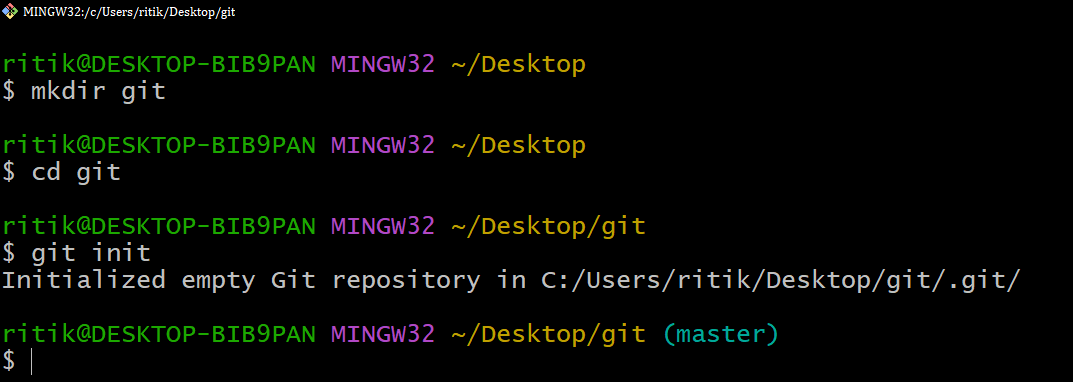
**Theory** **:** Git is a widely used open-source software tracking application used to track projects across different teams and revision level. It is a version control tool. To install Git follow the following procedure :

1. Browse to the official Git website: [https://git-scm.com/downloads](https://git-scm.com/downloads" \t "_blank)
2. Click the download link for Windows and allow the download to complete.
3. Browse to the download location (or use the download shortcut in your browser). Double-click the file to extract and launch the installer.
4. Allow the app to make changes to your device by clicking Yes on the User Account Control dialog that opens.
5. Review the GNU General Public License, and when you’re ready to install, click Next.
6. The installer will ask you for an installation location. Leave the default, unless you have reason to change it, and click Next.
7. A component selection screen will appear. Leave the defaults unless you have a specific need to change them and click Next.
8. The installer will offer to create a start menu folder. Simply click Next.
9. Select a text editor you’d like to use with Git and click Next.
10. The next step allows you to choose a different name for your initial branch. The default is ‘master.’ Unless you’re working in a team that requires a different name, leave the default option and click Next.
11. This installation step allows you to change the PATH environment. The PATH is the default set of directories included when you run a command from the command line. Leave this on the middle (recommended) selection and click Next.
12. The installer now asks which SSH client you want Git to use. Git already comes with its own SSH client, so if you don’t need a specific one, leave the default option and click Next.
13. The next option relates to server certificates. Most users should use the default. If you’re working in an Active Directory environment, you may need to switch to Windows Store certificates. Click Next.
14. The next selection converts line endings. It is recommended that you leave the default selection. This relates to the way data is formatted and changing this option may cause problems. Click Next.
15. Choose the terminal emulator you want to use. The default MinTTY is recommended, for its features. Click Next.
16. The installer now asks what the git pull command should do. The default option is recommended unless you specifically need to change its behavior. Click Next to continue with the installation.
17. Next you should choose which credential helper to use. Git uses credential helpers to fetch or save credentials. Leave the default option as it is the most stable one, and click Next.
18. Once the installation is complete, tick the boxes to view the Release Notes or Launch Git Bash, then click Finish.

**Git Repository :** Git Repository is a collection of all project files along with their history. It is a virtual storage of your project where you keep all the resources/files of the project along with a special folder called .git. The .git folder in a git repository is used by GIT programs to store information about the repository like Logs, Position of Head, and more. It allows you to save versions of your code, which can be accessed, tracked, and managed.

Steps to make git Repository :

1. Create a New Project/Folder. Command to create a folder on a Windows system is mkdir <folder name>.
2. Browse to New Project. Navigate to the project created in the previous step using the command cd <folder name>.
3. Initialize Bare Git Repository for the Project. Enter the command git init this command is used to Create Git Repository.

****

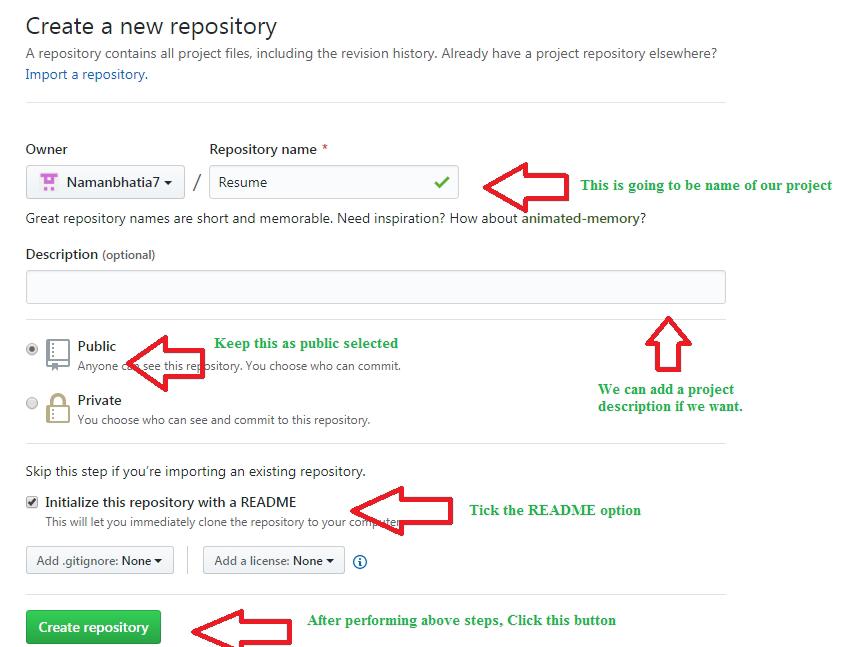
**Figure 1.** Initializing git repository

***Experiment 2***

**Aim :** Creating first GitHub Repository.

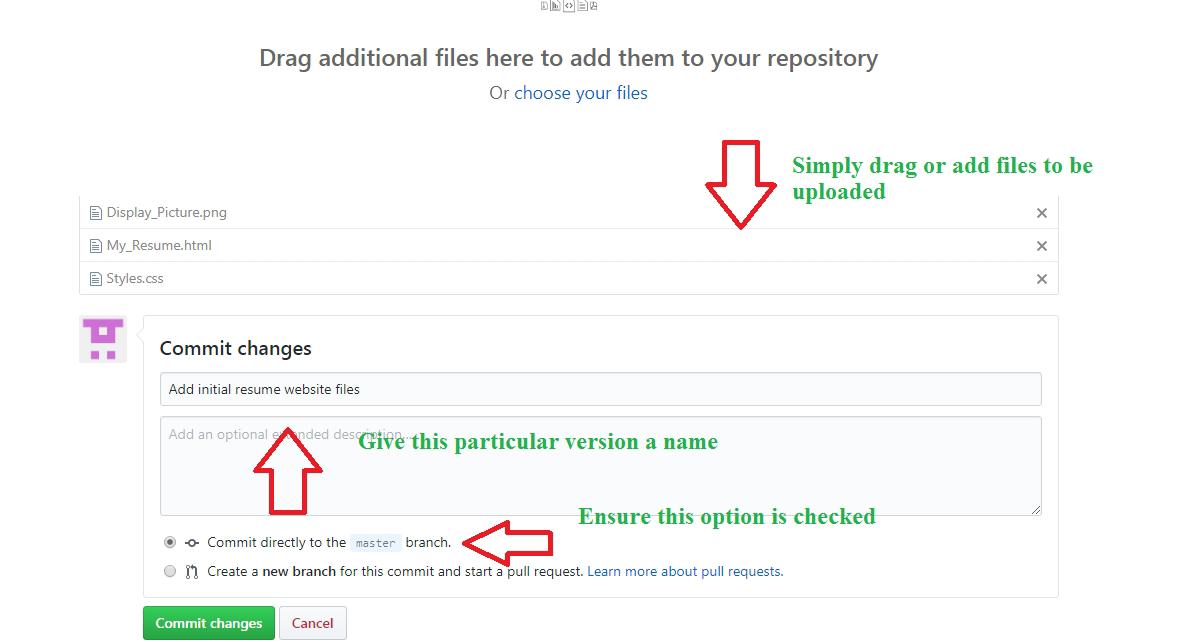
**Theory :** Introduction to GitHub: Git is an open-source version control system. It means that whenever a developer develops some project (like an app) or something, he/she constantly update it catering to the demands of users, technology and whatsoever it maybe. Version control systems keep these revisions straight, storing the modifications in a central repository. It allow developers to easily collaborate, as they can download a new version of the software, make changes, and upload the newest revision. Every developer can see these new changes, download them, and contribute. To create a Git Repository follow the following steps :

1. After successfully setting up GitHub account login to your account. Click on the new repository option.
2. After clicking new repository option, we will have to initialize some things like, naming our project, choosing the visibility etc. After performing these steps click Create Repository button.



**Figure 2.1** Creating GitHub repository

1. After clicking the button, we will be directed to below page. Right now the only file we have is a readme file.
2. Now click on the “Upload files” button.
3. Follow the steps mentioned in the Picture below and click “commit changes”.



**Figure 2.2** To commit changes to the project file/s.

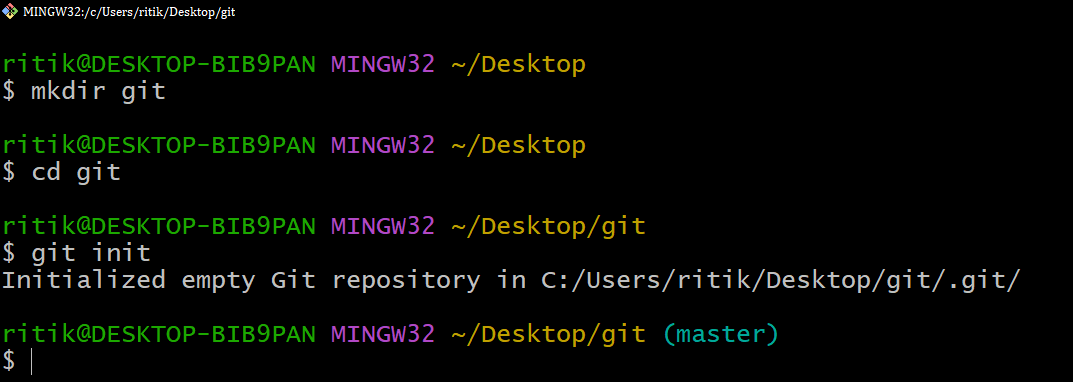
1. Now you will see that all of our files uploaded in our GitHub.

***Experiment 3***

**Aim :** Run command to initialize a repository on Git Bash.

**Theory :** The git init command creates a new Git repository. It can be used to convert an existing, unversioned project to a Git repository or initialize a new, empty repository.

1. Open Git Bash by right click and select Git Bash.
2. Make a directory by using mkdir <folder Name> in Git Bash.
3. Change the directory by using cd <folder Name>.
4. Now type git init to initialize directory as a Git Repository.



**Figure 3.1** Creating a git repository on local system

***Experiment 4***

**Aim :** Run command to add repository on Git Bash.

**Theory :**

***Experiment 5***

**Aim :** Run command to perform changes in repository on Git Bash.

**Theory :**

***Experiment 6***

**Aim :** Run command to create clone of GitHub remote repository on local system.

**Theory :**

***Experiment 7***

**Aim :** Installation of JDK and Eclipse IDE.

**Theory :** Steps to install JDK are as follows :

1. Download JDK from Oracle site.
2. Double click on the downloaded file.
3. Click Next to proceed with the installation.
4. Select the PATH to install Java in Windows.
5. Follow the onscreen instruction. Click close button once the installation is finished.
6. Right click o the This PC and select Properties.
7. Click on Advance Settings.
8. Click on Environment Variables to set Java runtime environment.
9. Click on New Button of User variables.
10. Type PATH in the Variable name.
11. Type the path, of bin folder which is install in JDK folder, in Variable Value and click OK.
12. Follow similar process to set the CLASSPATH.
13. Go to command prompt and type javac.

***Experiment 8***

**Aim :** Setting up Environment Variable for JAVA.

**Theory :**

***Experiment 9***

**Aim :** Create simple JAVA program and run using CLI.

**Theory :** To create java file follow the following steps :

1. Open notepad and write your program.
2. Save the file with extension .java.

And your java program is created.

javac <file\_name> command is used to compile the java code and java <name> is used to execute the code. Following are the steps to compile and run the java code :

1. First navigate to the folder where your java file is stored using cd <directory\_name> command.
2. Type javac <file\_name>, to compile your code.
3. If no error message appear then type java <name> to execute the code.

***Experiment 10***

**Aim :** Create MAVEN Project in Eclipse.

**Theory :**

***Experiment 11***

**Aim :** Test MAVEN Project using Junit.

**Theory :**

***Experiment 12***

**Aim :** Installation of Docker.

**Theory :** Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications. By taking advantage of Docker’s methodologies for shipping, testing, and deploying code quickly, you can significantly reduce the delay between writing code and running it in production. Following are the steps to install Docker :

1. Go to the website [https://docs.docker.com/docker-for-windows/install/](https://docs.docker.com/docker-for-windows/install/" \o "https://docs.docker.com/docker-for-windows/install/" \t "https://www.simplilearn.com/tutorials/docker-tutorial/_blank) and download the Docker File.
2. Then, double-click on the Docker Desktop Installer.exe to run the installer.
3. Note: Suppose the installer (Docker Desktop Installer.exe) is not downloaded; you can get it from Docker Hub and run it whenever required.
4. Once you start the installation process, always enable Hyper-V Windows Feature on the Configuration page.
5. Then, follow the installation process to allow the installer and wait till the process is done.
6. After completion of the installation process, click Close and restart.

***Experiment 13***

**Aim :** Run various command on Docker.

**Theory :** We use Docker for development, testing, and deployment because it’s easy to manage the environment. Because of its portability, it has become popular in the software industry. That mean’s if your code works in development, it will work in production. Moreover, developers can collaborate faster without worrying about which software dependency they need to install. Following are some Docker commands :

|  |  |  |
| --- | --- | --- |
| 1. No. | Command | Use |
| 1 | docker search <name> | To search public image on the Docker hub. It returns image name, description, stars, official and automated. |
| 2 | docker pull <name> | To pull the image we search from Docker hub. |
| 3 | docker image | List the images in our local system. |
| 4 | docker run <tags> | To create a container. |
| 5 | docker ps | List all the running container. |
| 6 | docker stop <container\_id> | To stop a container. |
| 7 | docker restart <container\_id> | To restart the stopped container. |
| 8 | docker rename <old\_name> <new\_name> | To change name of the container. |
| 9 | docker exec <container\_name> | To access the running container name. |
| 10 | docker logs <container\_name> | To debug Docker container. |
| 11 | docker rm <container\_name> | To remove a container. |
| 12 | docker rmi <image\_id> | To remove an image. |

***Experiment 14***

**Aim :** Case study of DevOps Tool : 1) Jenkins 2) Puppet 3)Nagios.

**Theory :**

1. Jenkins : **Jenkins** is an [open source](https://en.wikipedia.org/wiki/Open_source" \o "Open source) automation server. It helps automate the parts of [software development](https://en.wikipedia.org/wiki/Software_development" \o "Software development) related to [building](https://en.wikipedia.org/wiki/Software_build" \o "Software build), [testing](https://en.wikipedia.org/wiki/Test_automation" \o "Test automation), and [deploying](https://en.wikipedia.org/wiki/Software_deployment" \o "Software deployment), facilitating [continuous integration](https://en.wikipedia.org/wiki/Continuous_integration" \o "Continuous integration) and [continuous delivery](https://en.wikipedia.org/wiki/Continuous_delivery" \o "Continuous delivery). It is a server-based system that runs in [servlet containers](https://en.wikipedia.org/wiki/Java_Servlet" \l "Container_servers" \o "Java Servlet) such as [Apache Tomcat](https://en.wikipedia.org/wiki/Apache_Tomcat" \o "Apache Tomcat). It supports [version control](https://en.wikipedia.org/wiki/Version_control" \o "Version control) tools, including [AccuRev](https://en.wikipedia.org/wiki/AccuRev_SCM" \o "AccuRev SCM), [CVS](https://en.wikipedia.org/wiki/Concurrent_Versions_System" \o "Concurrent Versions System), [Subversion](https://en.wikipedia.org/wiki/Subversion_(software)" \o "Subversion (software)), [Git](https://en.wikipedia.org/wiki/Git_(software)" \o "Git (software)), [Mercurial](https://en.wikipedia.org/wiki/Mercurial" \o "Mercurial), [Perforce](https://en.wikipedia.org/wiki/Perforce" \o "Perforce), [ClearCase](https://en.wikipedia.org/wiki/ClearCase" \o "ClearCase) and [RTC](https://en.wikipedia.org/wiki/Rational_Team_Concert" \o "Rational Team Concert), and can execute [Apache Ant](https://en.wikipedia.org/wiki/Apache_Ant" \o "Apache Ant), [Apache Maven](https://en.wikipedia.org/wiki/Apache_Maven" \o "Apache Maven) and [sbt](https://en.wikipedia.org/wiki/Sbt" \o "Sbt) based projects as well as arbitrary [shell scripts](https://en.wikipedia.org/wiki/Shell_script" \o "Shell script) and Windows [batch commands](https://en.wikipedia.org/wiki/Batch_file" \o "Batch file).
2. Puppet : Puppet is an open source software configuration management and deployment tool. It's most commonly used on Linux and Windows to pull the strings on multiple application servers at once. But you can also use Puppet on several platforms, including [IBM mainframes](http://www.zdnet.com/article/puppet-devops-comes-to-the-mainframe/" \t "https://www.hpe.com/us/en/insights/articles/_blank), Cisco switches, and Mac OS servers. Like other DevOps programs, Puppet does more than automate system administration. It changes the human workflow, and enables developers and system administrators to work together. Programmers can write, test, and launch applications without waiting on Ops staff to deliver the resources needed.

Nagios : Nagios is an open source monitoring system for computer systems. It was designed to run on the Linux operating system and can monitor devices running Linux, Windows and Unix operating systems. Nagios software runs periodic checks on critical parameters of application, network and server resources. For example, Nagios can monitor memory usage, disk usage, microprocessor load, the number of currently running processes and log files. Nagios also can monitor services, such as Simple Mail Transfer Protocol (SMTP), Post Office Protocol 3 (POP3), Hypertext Transfer Protocol (HTTP) and other common network protocols. Active checks are initiated by Nagios, while passive checks come from external applications connected to the monitoring tool.

***Experiment 15***

**Aim :** Write case study on various Agile Models.

**Theory :** Agile is an iterative and incremental approach to project management. Agile team works in iterative cycles and can modify deliverable quickly to suit customer’s feedback. It is best for projects which requires extreme flexibility and speed. It focuses on customer’s satisfaction than documentation. It is highly flexible however it becomes difficult to predict the resources needed since end result is likely to change based on customer’s feedback. Following are the various Agile Model :

* Scrum : It has an iterative project management style. It follows agile principle. In scrum, work is done in sessions a.k.a. sprints and scrum master leads the project team. In sprints, team members meet regularly, discuss work together and tackle obstacles. Scrum is good in rapid development and testing therefore deliverable are usually on time. But, daily meeting can be time consuming and may not everybody’s schedule.
* Kanban :Kanban focuses on individual team capacity. Teams creates visual tasks using whiteboard or custom dashboard and evenly distributes the workload between the teams. Kanban offers a bird’s eye view of all the task in the project making it easier to track the progress. However, their are unclear dates and deadline since, their is no established timeline.
* Extreme Programming (XP) : Extreme Programming (XP) focuses on customer. It aims to improve software quality by changing work sprint to suit customer’s needs and improve quality. In XP, changes can be made quickly. However, due to that it become hard to track update of the project.
* Lean : Lean is an agile framework. It uses a work breakdown process to spot potential waste areas and cut them out leaving only the essentials. It saves resources needed for the project but it may involve low levels of inventory which is risky.

***Index :***

|  |  |  |  |
| --- | --- | --- | --- |
| *S.No.* | *Experiment* | *Page no.* | *Teacher sign* |
| *1* | *Installation of Git and creating Git Repository* | *1 - 3* |  |
| *2* | *Creating first GitHub Repository* | *3 - 5* |  |
| *3* | *Run command to initialize a repository on Git Bash* | *5 - 6* |  |
| *4* | *Run command to add repository in Git Bash.* | *6 -* |  |
| *5* | *Run command to perform changes in repository on Git Bash.* |  |  |
| *6* | *Run command to create clone of GitHub remote repository on local system* |  |  |
| *7* | *Installation of JDK and Eclipse IDE.* |  |  |
| *8* | *Setting up Environment Variables for JAVA.* |  |  |
| *9* | *Create simple JAVA program and run using CLI.* |  |  |
| *10* | *Create MAVEN Project in Eclipse.* |  |  |
| *11* | *Test MAVEN Project using Junit.* |  |  |
| *12* | *Installation of Docker* |  |  |
| *13* | *Run various commands on Docker.* |  |  |
| *14* | *Case study of DevOps tool : 1) Jenkins 2) Puppet 3) Nagios* |  |  |
| *15* | *Write a case study of various Agile Models.* |  |  |