# INDEX

Ex. NO : 1

24.02.22

# CREATE AND MANAGE LOCAL REPOSITORY USING GIT

## AIM :

To create and manage local repository using GIT.

## ALGORITHM :

Step 1 : Download git from your browser and install it.

Step 2 : In desktop, right click and select Git bash here.

Step 3 : This opens the git bash, there create a new repository using mkdir command and to set your Git email, run the git config - global user. email command.

Step 4 : To initiate the created repository, use git init command.

Step 5 : Use touch command to create three empty files that are html file, text file and python file.

Step 6 : To add all the three files to the index (staging Area) use the git add. command.

Step 7 : Using git commit -m "value" command we can move all the files from index to a commit.

Step 8 : By the use of git status command we can display the state of the repository and staging area.

Step 9 : Use git log command to review and read a history of everything that happens to a repository but if you want to specifically see for a file use git log --oneline file-name.

Step 10 : The git branch command allows you to create, list, rename and delete branches and the git checkout command lets you navigate between the branches created by git branch.

RESULT :

Hence a local repository has been created and managed using GIT.

Ex. No: 2

04.03.22

# REMOTE REPOSITORY

## AIM:

To create and manage a remote repository using GIT.

## ALGORITHM:

Step 1: Sign up to your git hub account and click the plus option on the top right then click new repository.

Step 2: Fill in your repository name and description then select public. Also it is recommended to select add a readme file, add.gitignore and choose a license and click on create repository.

Step 3: In your repository start either by creating a new file uploading an existing file.

Step 4: You can create or upload many versions of your file and commit the changes.

Step 5: By using fork we can copy the entire public repository of any other users in to our account also fork make changes to a project without affecting the original repository.

Step 6: To use fork login with another account and copy and Paste URL of repository then just click on fork to clone to others account. Suppose we want to fork public repository "calc-log". So, search for "calc_log" github repository on google and once it is opened click on "Fork button" from the top of the github web page.

Step 7: After fork it will be added on your local repo -sitory and you can check it from Insights tab.

Step 8: To delete the repository, open the desired repos -itory you want to delete and go to the settings option. There you will see delete repository button to delete it.

Step 9: Now, if you want to download a repository on local machine, then git clone command is used followed by path to repository. In GitHub the path of repository can be known through clone or download button.

RESULT :

Hence a remote repository has been created and managed using GIT.

Ex. No: 3

11.03.22

# CREATE AND PUSH A FILE FROM LOCAL REPOSITORY TO REMOTE REPOSITORY

## AIM:

To create and push a file from local repository to remote repository in git.

## ALGORITHM:

Step 1: Right click on your desktop and select git bash to open the git bash.

Step 2: To create a repository use the mkdir followed by the name of the repository, for example, "mkdir devops7".

Step 3: Get into the repository by using the command "cd devops7".

Step 4: Using the command "git init" initialize the directory you created.

Step 5: Create a file using the command "touch index.html" and check if it is staged or not using the "git status" command.

Step 6: To add values to the created file use "nano index.html" command which will open the nano editor, there you can type the html code.

Step 7: To save the file press Ctrl + O and press enter. To exit the nano editor press Ctrl + X.

Step 8: To list the content inside the created file use "cat index.html" command and stage this file using "git add index.html" command.

Step 9: To commit this file to the local repository type "git commit -m" Added index.html".

Step 10: By using the "git status" command again check the status of the file.

Step 11: Now to push the file to the remote repository sign in to your repository and copy the https link and go back to the terminal.

Step 12: To add this repository as an origin location use "git remote add origin https://github.com/PavithranS/Calc_Pavi.git" where https://github.com/PavithranS/Calc_Pavi.git is the link of your remote repository.

Step 13: Using the "git push origin master" command you can push the file to the remote repository.

Step 14: If verification needed re-enter the code given

and verify your account.

Step 15: Now locate the remote repository and reload the page where a new branch called master would be added.

Step 16: Change the branch from main to master by clicking the drop down box and there you can see the index.html file.

RESULT:

Hence a file is created and pushed from local to remote repository successfully.

| EX. NO: 4 | CREATE FIRST JOB IN JENKINS |
|---|---|
| 19.08.22 | |

AIM:

To create the first job in Jenkins and build it using simple windows batch commands.

ALGORITHM:

Step 1: Browse to 'http://localhost:8080' to unlock Jenkins.

Step 2: Login to Jenkins using your login credentials.

Step 3: Click on New Item in the Jenkins dashboard page to create a new job.

Step 4: Give a unique name for your project (job) in the 'Enter an item name' field.

Step 5: Select the job type as 'Freestyle Project' listed below the name field and click OK.

Step 6: In the General tab of the configuration page give the description of your project.

Step 7: Click on Advanced option to give optional display name that is to be shown for the project throughout the Jenkins web GUI.

Step 8: Now move to the Build tab either by scrolling

down or by clicking the options from the top and select Add build step option.

p 9: In the Add build step option choose 'Execute Windows batch command' to run a windows batch script for building a project.

ep 10: We will build the job by checking the java version in the local system and by printing the "Hello" text using the following commands:

- check Java version : java - version
- print the statements : echo "Hello! Welcome to Jenkins"

ep 11: Now click Apply and then click save.

p 12: The created project will appear in the workspace with the given project name, Display name and Description.

p 13: To build the project select on 'Build Now' option which is displayed to left of workspace.

14: You will find the completed build run below the 'Build History' in the left, the first build is indicated as '#1'.

15: Click on the '#1' and select 'Console Output'.

tep 16: The console output displays whether the build is successful or not.

ESULT:

The job has been created and built using uild tab. Hence, the output is obtained successfully nd the result is displayed.

Ex. No: 5

05.04.22

# CONFIGURE EMAIL NOTIFICATIONS IN JENKINS

## AIM:

To configure email notifications on Jenkins using the mail plugins.

## ALGORITHM:

Step 1: Open the Chrome browser, select your profile and click on 'Manage your Google Account'.

Step 2: In the my accounts page select 'Security' and scroll down to click on 'App passwords' for generating passwords.

Step 3: Now give your Gmail account password and select next.

Step 4: In the App passwords console, select the app and device for password generation and then click on Generate.

Step 5: Copy the Generated App Password and select done.

Step 6: Now browse to 'http://localhost:8080' to unlock Jenkins.

Step 7: Login to Jenkins using your login credentials.

Step 8: In the Jenkins Dashboard, Select on 'Manage Jenkins' and then click 'Configure System' in the system configuration.

Step 9: Scroll down to Extended Email Notification and give 'smtp.gmail.com' in the SMTP server field and in the SMTP port field enter '465' then click on Advanced.

Step 10: Leave the Credentials field default and then select 'Use SSL'.

Step 11: Then scroll the 'Advanced Email Properties' and move to the 'Default user e-mail suffix' field and enter the email suffix as '@gmail.com'.

Step 12: Leave the other fields empty and scroll down to 'Additional groovy classpath' check on the 'Enable debug mode'. Click on 'Default Triggers'.

Step 13: Check on 'Always' in the below listed check boxes and then move to Email notification.

Step 14: Again in the SMTP server field enter 'smtp.gmail.com' and in the Default user e-mail suffix field enter '@gmail.com'. Then click on Advanced.

Step 15 : Check on the 'Use SMTP Authentication' and enter your Gmail Id and password on the 'Username and Password' fields.

Step 16 : Now Check on 'Use SSL' field and enter the SMTP port as '465'.

Step 17 : Check on the 'Test configuration by sending test e-mail' field under the 'Test e-mail recipient' enter your mail id and then click on 'Test configuration'.

Step 18 : You will see a notification as 'Email was successfully sent'. Now click Apply and then click save.

Step 19 : In order to view the sample test email open your mail inbox.

RESULT!

Thus the email notification configuration has been completed and the email has been successfully delivered.