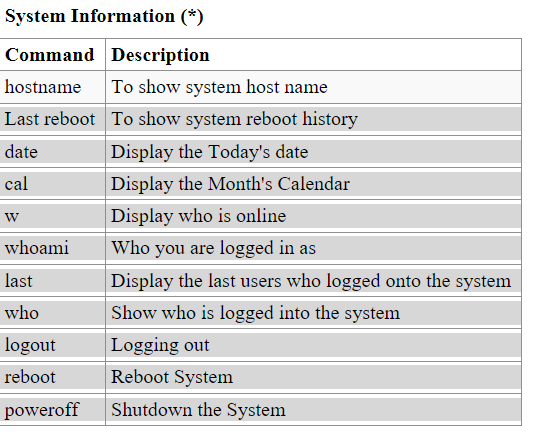
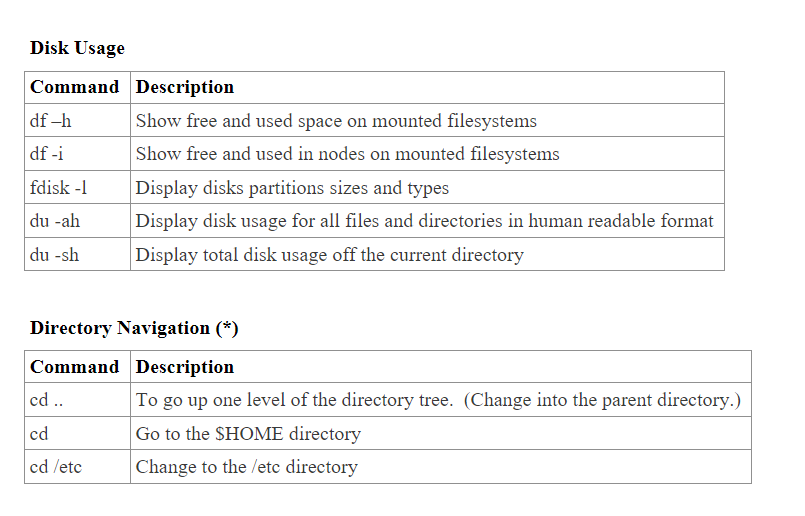
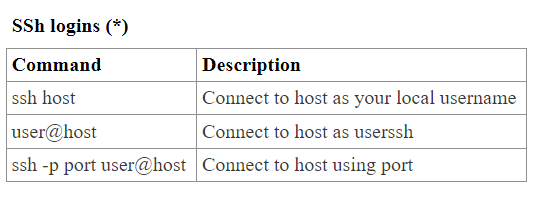
LINUX COMMANDS



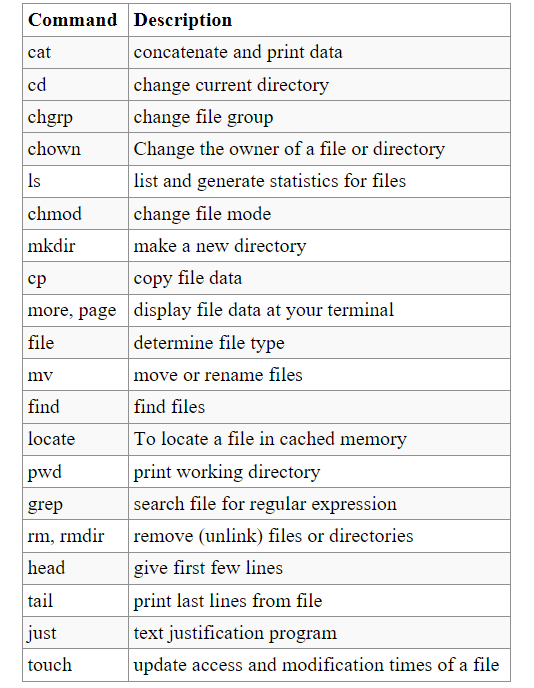


File Permissions :

r-4,w-2,x-1

to check ls -l

Execute 
Write 
—rwxrw—r— 
User Group Other 
File type: 
regular file 
All 
d directory 
PERMISSION 
EXAMPLE 
gbugd 777 filename 
rwx rwx rwx 
775 filename 
rwx r-x r-x chmod 755 filename 
(x) 
rw- r-- r— 
ghmod 664 filename 
chmod 644 filename 
# NOTE: Use 777 sparingly! 
LEGEND 
U = User 
G = Group 
W = World 
r Read 
w = write 
tel 
x = execu 
- = no access 



CI-CD

BlJlLD 
CONTlNlJOlJS 
INTEGRATION 
TEST 
MERGE 
CONTlNUOlJS 
DELIVERY 
AlJTOMATlCALLY 
RELEASE ΤΟ 
REPOSITORY 
CONTlNUOUS 
DEPLOYMENT 
AlJTOMATlCALLY 
DEPLOY ΤΟ 
PRODUCTION 

CI/CD bridges the gaps between development and operation activities and teams by enforcing automation in building, testing and deployment of applications.

CI/CD services compile the incremental code changes made by developers, then link and package them into software deliverables.

Automated tests verify the software functionality, and automated deployment services deliver them to end users.

The aim is to increase early defect discovery, increase productivity, and provide faster release cycles.

The process contrasts with traditional methods where a collection of software updates were integrated into one large batch before deploying the newer version. Modern-day.

DevOps practices involve continuous development, continuous testing, continuous integration, continuous deployment and continuous monitoring of software applications throughout its development life cycle.

The CI/CD practice, or CI/CD pipeline, forms the backbone of modern day DevOps operations.

code 
ope 

What is Jenkins ?

Jenkins is a powerful application that allows continuous integration and continuous delivery of projects, regardless of the platform you are working on. It is a free source that can handle any kind of build or continuous integration. You can integrate Jenkins with a number of testing and deployment technologies. In this tutorial, we would explain how you can use Jenkins to build and test your software projects continuously.

JIRA

**Jira** is a proprietary issue tracking product developed by Atlassian that allows bug tracking and agile project management.

The product name is a truncation of *Gojira*, the Japanese word for Godzilla. The name originated from a nickname Atlassian developers used to refer to Bugzilla, which was previously used internally for bug-tracking.

According to Atlassian, Jira is used for issue tracking and project management by over 180,000 customers in 190 countries. Some of the organizations that have used Jira at some point in time for bug-tracking and project management include Fedora Commons, Hibernate, and the Apache Software Foundation, which uses both Jira and Bugzilla. Jira includes tools allowing migration from competitor Bugzilla.

Jira is offered in four packages:

* **Jira Work Management** is intended as generic project management.
* **Jira Software** includes the base software, including agile project management features (previously a separate product: Jira Agile).
* **Jira Service Management** is intended for use by IT operations or business service desks.
* **Jira Align**is intended for strategic product and portfolio management

Jira is written in Java and uses the Pico inversion of control container, Apache OFBiz entity engine, and WebWork 1 technology stack. For remote procedure calls (RPC), Jira has REST, SOAP, and XML-RPC interfaces. Jira integrates with source control programs such as Clearcase, Concurrent Versions System (CVS), Git, Mercurial, Perforce, Subversion, and Team Foundation Server. It ships with various translations including English, French, German, Japanese, and Spanish.

Jira implements the Networked Help Desk API for sharing customer support tickets with other issue tracking systems.

Confluence

 Confluence is a collaborative documentation tool, while Jira is primarily a project management and issue tracking software. They can be integrated with each other and used together, or they can be used separately.

VERSION CONTROL

Version control (also known as revision control, source control, or source code management) is a class of systems responsible for managing changes to computer programs, documents, large web sites, or other collections of information. Version control is a component of software configuration management.

These systems are most commonly run as stand-alone applications, but revision control is also embedded in various types of software, such as word processors and spreadsheets, collaborative web docs, content management systems, e.g., Wikipedia's page history.

Why use version control?

Software is developed to solve a user problem. Increasingly, these solutions have many different forms (e.g. mobile, embedded, SaaS) and run a variety of environments, such as cloud, on-prem, or Edge.

The purpose of version control is to allow software teams track changes to the code, while enhancing communication and collaboration between team members. Version control facilitates a continuous, simple way to develop software.

What are the types of version control systems?

The two most popular types of version control systems are **centralized** and **distributed**. Centralized version control systems store all the files in a central repository, while distributed version control systems store files across multiple repositories. Other less common types of version control systems include lock-based and optimistic.

1. Distributed version control system
2. Centralized version control system
3. Lock-based version control system
4. Optimistic version control system

Benefits of version control

Quality

Teams can review, comment, and improve each other’s code and assets.

Acceleration

Branch code, make changes, and merge commits faster.

Visibility

Understand and spark team collaboration to foster greater release build and release patterns.

GIT

Git is a free and open-source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

**GitHub:** GitHub is a repository hosting service tool that features collaboration and access control. It is a platform for programmers to fix bugs together and host open-source projects. GitHub is designed for the developers and to help them track their changes into a project through the repository.

Following are some features of GitHub:

·       Specifies milestones and labels to the projects.

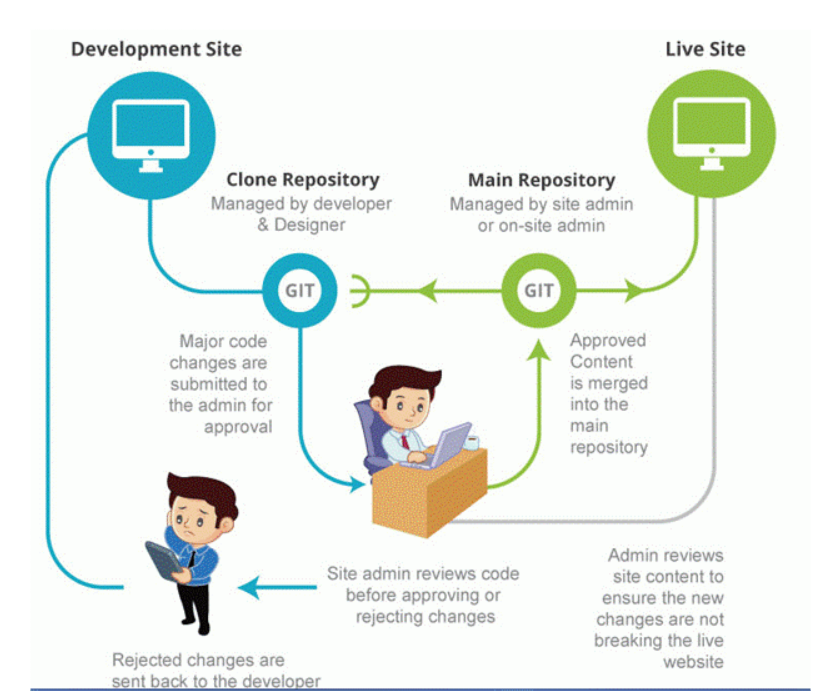
·       Comparison view between branches is allowed.

·       GitHub Pages allows us to publish and host websites within GitHub.

·       Syntax highlight feature.

·       It allows third-party API integrations for bug tracking and cloud hosting.

<https://docs.google.com/document/d/15SHPhkzsSzgqbOQSHC4IOBwE4TttPbrH/edit>



SDLC

Software Development Life Cycle is a process used by the software industry to design, develop and test high quality software. The SDLC aims to produce a high-quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates.

A typical Software Development Life Cycle consists of the following stages −

Project Manager -

Stage 1: Planning and Requirement Analysis

Stage 2: Defining Requirements

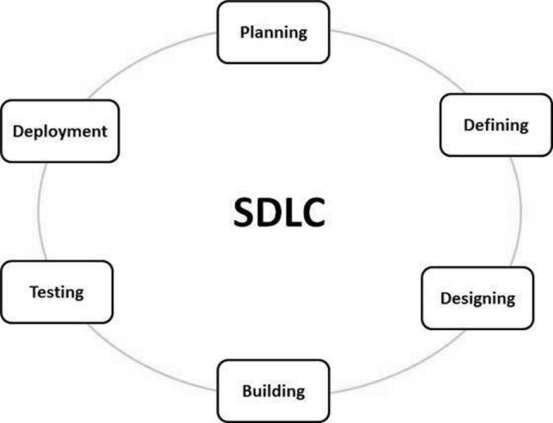
Stage 3: Designing the Product Architecture

Stage 4: Building or Developing the Product

Stage 5: Testing the Product

Stage 6: Deployment in the Market and Maintenance

SDLC is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process.



AGILE MODELING:

**Agile modeling** is a methodology for modeling and documenting software systems based on best practices. It is a collection of values and principles, that can be applied on an (agile) software development project. This methodology is more flexible than traditional modeling methods, making it a better fit in a fast changing environment. agile software development tool kit.

**Why Agile model is used?**

With Agile software development, teams can quickly adapt to requirements changes without negatively impacting release dates. Not only that, Agile helps reduce technical debt, improve customer satisfaction and deliver a higher quality product.

g 
Production & 
Technical 
Support 
. ostrom 
Agile 
Development 
Methodology 
nssy Ane 
Fig. Agile Model 
Design 
Document & 
Prototype 
terations , Oemo & 
Feedback 
terations, Demo 
& Feedback 