CaseStudy 7:

Task (1) Version control systems are a category of software tools that helps record changes to files by keeping a track of modifications done to the code.

Use of Version Control System:

1. A repository: It can be thought as a database of changes. It contains all the edits and historical versions (snapshots) of the project.
2. Copy of Work (sometimes called as checkout): It is the personal copy of all the files in a project. You can edit to this copy, without affecting the work of others and you can finally commit your changes to a repository when you are done making your changes.

Types of Version Control Systems:

1. Local Version Control Systems
2. Centralized Version Control Systems
3. Distributed Version Control Systems

Purpose of Version Control:

1. Multiple people can work simultaneously on a single project.
2. It also enables one person to use multiple computers to work on a project.
3. It integrates the work that is done simultaneously by different members of the team.
4. Version control provides access to the historical versions of a project.

Task (2)

Version control is a system that records changes to a file or set of files over time so that can recall specific versions later.

We need version control system to keep track of changes and keep every team member working off the latest version. So, should use version control software for all code, files, and assets that multiple team members will collaborate on. It needs to do more than just manage and track files.

Task (3)

Now we use git version control system.

What is Git?

Git is a distributed version-control system for tracking changes in source code during software development.

Git is designed for coordinating work among programmers, but it can be used to track changes in any set of files.

Its goals include speed, data integrity, and support for distributed, non-linear workflows.

Here are some common commands for using Git:

git init: initializes a brand new Git repository and begins tracking an existing directory.

git clone: creates a local copy of a project that already exists remotely. The clone includes all the project's files, history, and branches.

git add: is command performs staging, the first part of that two-step process. Any changes that are staged will become a part of the next snapshot and a part of the project's history. Staging and committing separately gives developers complete control over the history of their project without changing how they code and work.

git commit: saves the snapshot to the project history and completes the change-tracking process. In short, a commit functions like taking a photo. Anything that's been staged with git add will become a part of the snapshot with git commit.

git status: shows the status of changes as untracked, modified, or staged.

git branch: shows the branches being worked on locally.

git pull: updates the local line of development with updates from its remote counterpart. Developers use this command if a teammate has made commits to a branch on a remote, and they would like to reflect those changes in their local environment.

git push: updates the remote repository with any commits made locally to a branch.

Task (4)

Testing is a method to check whether the actual software product matches expected requirements and to ensure that software product is defect free.

It involves execution of software/system components using manual or automated tools to evaluate one or more properties of interest.

The purpose of software testing is to identify errors, gaps or missing requirements in contrast to actual requirements.

Testing is Important because if there are any bugs or errors in the software, it can be identified early and can be solved before delivery of the software product.

Properly tested software product ensures reliability, security and high performance which further results in time saving, cost effectiveness and customer satisfaction.

Testing is important because software bugs could be expensive or even dangerous.

Task (5)

JUnit is a unit testing framework for the Java programming language.

JUnit has been important in the development of test-driven development, and is one of a family of unit testing frameworks which is collectively known as xUnit that originated with SUnit.

JUnit is linked as a JAR at compile-time; the framework resides under package junit.framework for JUnit 3.8 and earlier, and under package org.junit for JUnit 4 and later.