**Software Engineering Tools Lab Assignment No-5**

(Module 4and5- Understanding version control using VSS and Managing code using SVN )

**Due date-06/03/2023**

Q 1. What is version control system and why it is important?

A version control system (VCS) is a software tool that helps developers manage changes to their source code, documents, and other files. VCS provides a way to track different versions of files, document changes made over time, and collaborate on shared codebases with other developers.

The main purpose of a VCS is to enable developers to work collaboratively on a codebase, without fear of overwriting each other's work or losing important changes. VCS allows developers to track changes to their code over time, compare different versions of a file, and revert to a previous version if necessary. This is especially useful when multiple people are working on the same codebase, or when a developer needs to revisit an older version of their code.

Q 2. Illustrate different types of version control system with example.

There are two main types of version control systems: centralized version control systems (CVCS) and distributed version control systems (DVCS).

1. Centralized Version Control System (CVCS): In a CVCS, there is a single central repository that stores the entire version history of a project. Developers check out files from this repository to work on them and then check them back in when they are done. Examples of CVCS include Subversion (SVN), CVS, and Perforce.
2. Example of CVS: Concurrent Versions System (CVS) is a popular centralized VCS that has been widely used in software development. It allows developers to work on a shared codebase and manage changes to the code over time. CVS stores all versions of files in a central repository and allows developers to check out files to make changes, then check them back in when they are done.
3. Distributed Version Control System (DVCS): In a DVCS, each developer has a complete copy of the repository on their own machine. Changes are made locally, and then pushed or pulled to and from other developers' repositories. This allows for a more decentralized workflow, with developers working independently and merging changes when necessary. Examples of DVCS include Git, Mercurial, and Bazaar.
4. Example of Git: Git is a popular DVCS that has become widely used in software development. Each developer has a complete copy of the repository on their own machine, and changes are made locally and pushed or pulled to and from other developers' repositories. Git allows developers to work independently and merge changes when necessary, making it a powerful tool for collaborative development. It is also widely used in open-source software development, with many projects hosted on platforms like GitHub and GitLab.

Q 3. Perform below operations using CVS

1. **cvs checkout**

cvs checkout project-name

cvs checkout myproject

1. **cvs update**

cd myproject

cvs update

1. **cvs add**

cd myproject

cvs add newfile.txt

1. **cvs remove**

cd myproject

cvs remove oldfile.txt

1. **cvs commit**

cd myproject

cvs commit -m "Added newfile.txt and removed oldfile.txt"

Q 4. Differentiate Between The Git & SVN Repository?

1. Distributed vs Centralized Repository: Git is a distributed version control system, meaning that each developer has a complete copy of the repository on their own machine, and changes can be made and committed locally. SVN, on the other hand, is a centralized version control system, where there is a single central repository that stores the entire version history of a project.
2. Branching and Merging: Git makes branching and merging changes between branches a simple process, allowing developers to work on multiple features or bug fixes simultaneously and merge changes back into the main branch when they are ready. SVN also supports branching and merging, but it can be more complex and requires more coordination among developers.
3. Speed: Git is generally faster than SVN, especially when it comes to large repositories or frequent branching and merging.
4. File Renaming and Moving: Git tracks files based on content, which means that renaming or moving a file within a repository is a simple operation. In SVN, however, renaming or moving a file can require copying and deleting, which can be more time-consuming and error-prone.
5. Conflict Resolution: Git has more sophisticated conflict resolution capabilities than SVN, which can make it easier for developers to resolve conflicts that arise when multiple developers make changes to the same code.

Q 5. What is “branch”, “tag” And “trunk” In SVN?

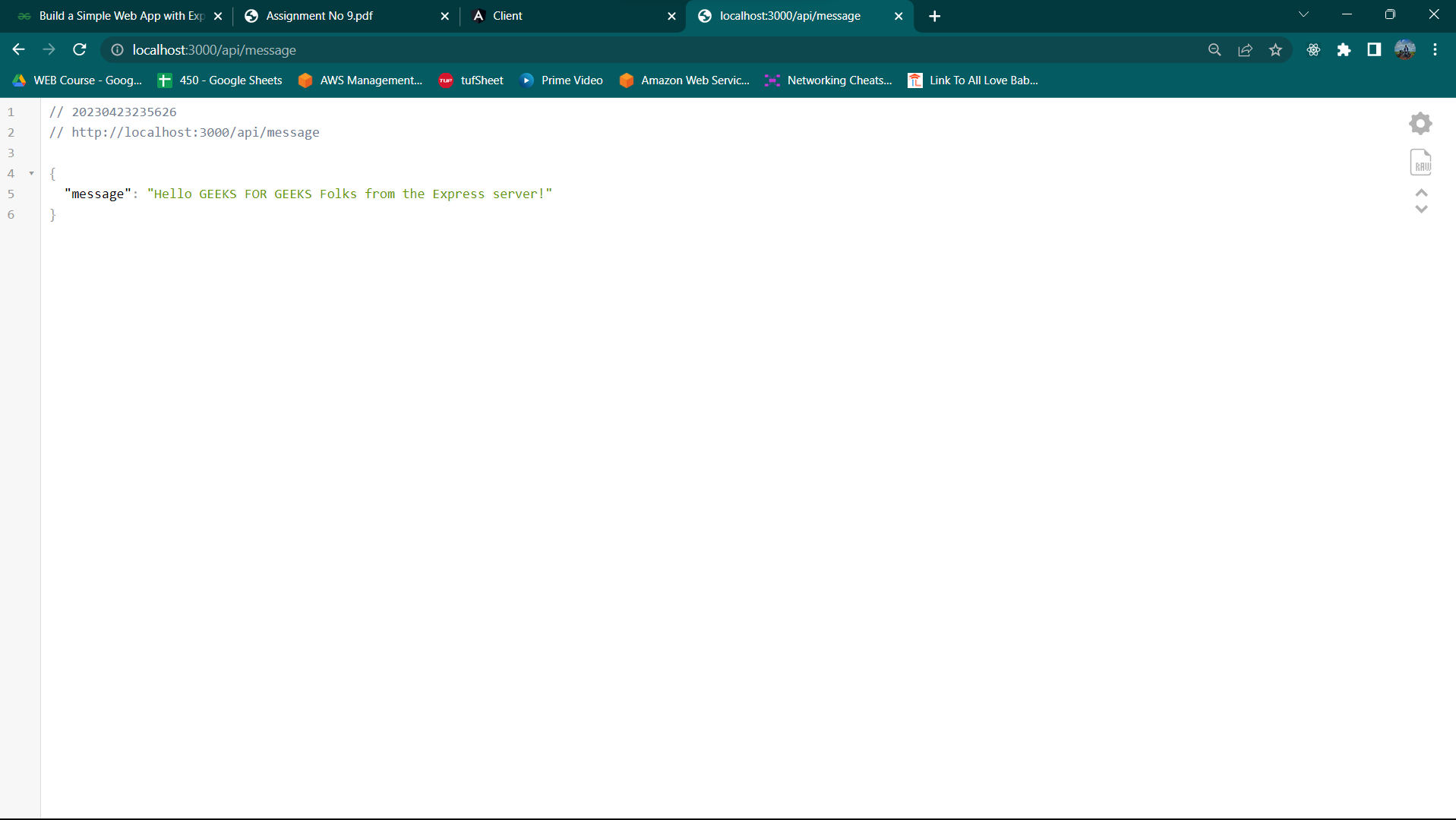
1. Branch: A branch is a copy of the codebase that is created to allow for parallel development. Branches can be used to work on new features, bug fixes, or other changes without affecting the main codebase. Once the changes on a branch are complete and tested, they can be merged back into the main codebase.
2. Tag: A tag is a way of marking a specific point in the codebase, typically a release or a significant milestone. Tags are useful for creating snapshots of the code at a particular point in time, and they can be used to revert to a specific version of the code if necessary.
3. Trunk: The trunk is the main line of development in the codebase, also known as the "master" branch. All development work is typically done on branches, and changes are merged back into the trunk when they are complete and tested.

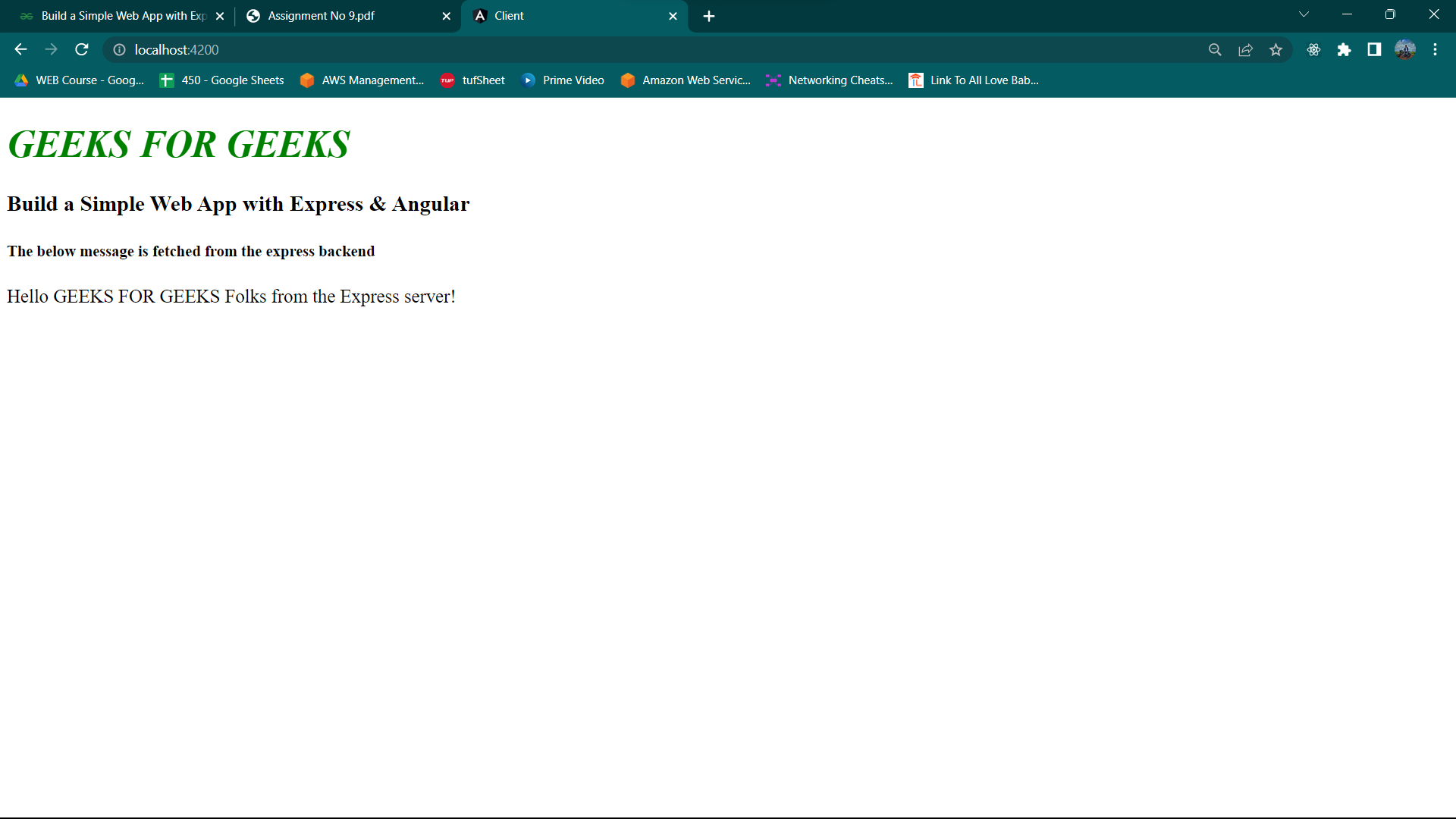
* The trunk is the main line of development, where the most up-to-date and stable version of the code is stored.
* Branches are created off the trunk to work on specific features or changes. Developers can work on the branch without affecting the trunk, and changes can be merged back into the trunk when they are complete.
* Tags are created to mark specific points in the codebase, such as a release or a milestone.

1. 6. How CVS is different from SVN?

* Centralized vs. Distributed: CVS is a centralized version control system, which means that there is a single central repository that stores the entire version history of the project. SVN, on the other hand, supports both centralized and distributed workflows, allowing users to work with a local copy of the repository.
* Branching and Merging: SVN has better support for branching and merging than CVS. In SVN, branches are cheap and easy to create, and merging changes between branches is generally straightforward. In contrast, CVS branches can be more complex to manage and merge, which can make it more difficult for developers to work on multiple features simultaneously.
* File Renaming and Moving: SVN tracks file moves and renames more efficiently than CVS, making it easier to track changes to files and directories.
* Speed: SVN is generally faster than CVS, particularly for larger repositories.
* Repository Backup: CVS requires additional software to perform backups, while SVN has built-in backup functionality.
  1. Demonstrate a display the app version in angular.
  2. Build a simple web app with Express and Angular.

Referance:https://www.geeksforgeeks.org/build-a-simple-web-app-with-expressangular/?ref=rp





* 1. What is git version control?
* Git is a distributed version control system that allows developers to track changes to code, collaborate with others, and manage code repositories. It was created by Linus Torvalds in 2005 and has since become one of the most widely used version control systems in the world.
* Git stores code changes as a series of snapshots, or "commits," which can be versioned, merged, and tracked over time. Developers can work on multiple versions of a codebase simultaneously, without affecting each other's work, and can merge changes between branches to keep the codebase up to date and functional.
* Git is designed to be fast, efficient, and flexible, and it supports a wide range of workflows, from simple personal projects to complex enterprise development. It is also highly customizable, with a wide range of third-party tools and integrations available to extend its functionality and support specific development needs.
* Overall, Git is an essential tool for modern software development, allowing developers to collaborate, manage, and track code changes in a flexible and efficient way.
  1. Demonstrate creation of repository in git.
* Open the terminal or command prompt on your computer and navigate to the directory where you want to create the repository.
* Type the following command to initialize a new Git repository:

**git init**

* You can now start adding files to the repository by creating new files in the directory or copying existing files into it.
* Once you have added files to the repository, use the following command to stage them for commit:

**git add .**

* Use the following command to commit the changes and create a new commit:

**git commit -m "Initial commit"**

* Your Git repository is now created and initialized, and you have committed the initial changes to it. You can continue to work on the project and make new commits as necessary, using Git to manage version control and collaboration.