Experiment no. 8

Aim: Change specification and use any SCM (Software Configuration Management) Tool to make different versions.

Tools:

- Git / GitHub / GitLab / Bitbucket
- IDE (VS Code, Eclipse, IntelliJ, PyCharm)
- Command Line / Git Bash
- Optional GUI tools: GitKraken, SourceTree

Theory:

Software Configuration Management (SCM) is a discipline of controlling and managing software changes. It ensures that modifications in code, documents, and requirements are systematically tracked and maintained.

Key aspects of SCM:

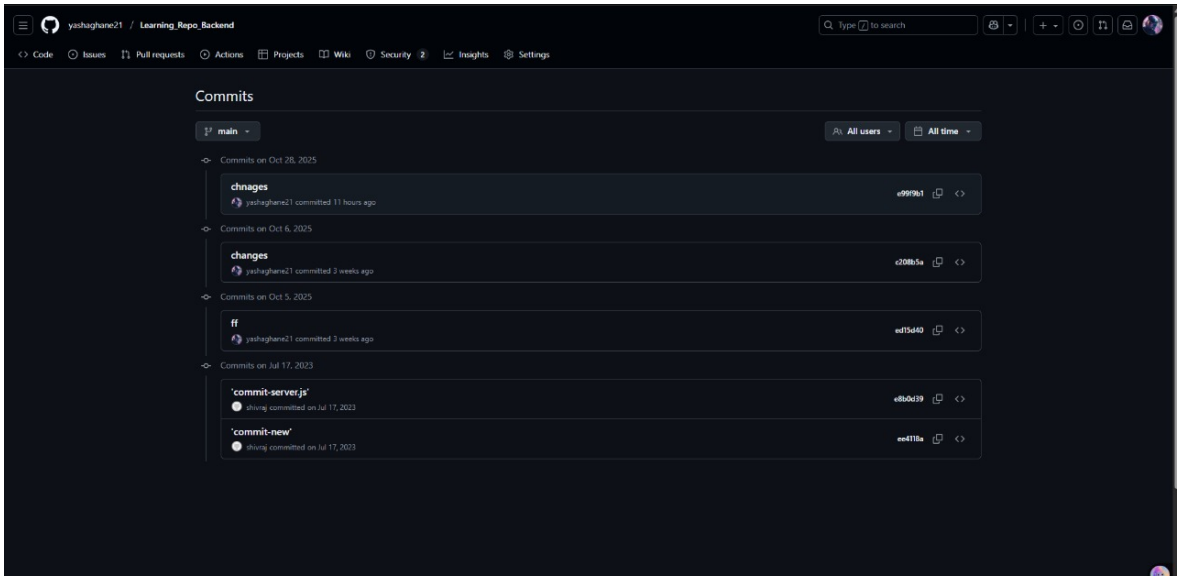
- Version Control – keeping multiple versions of code/specifications for traceability.
- Change Specification – clearly describing the modification made in a new version.
- Branching and Merging – creating separate lines of development and later integrating them.
- Baselines – reference points in development that are formally reviewed and agreed upon.
  By using tools like Git, teams can manage project evolution, roll back to stable versions, collaborate effectively, and maintain software quality.

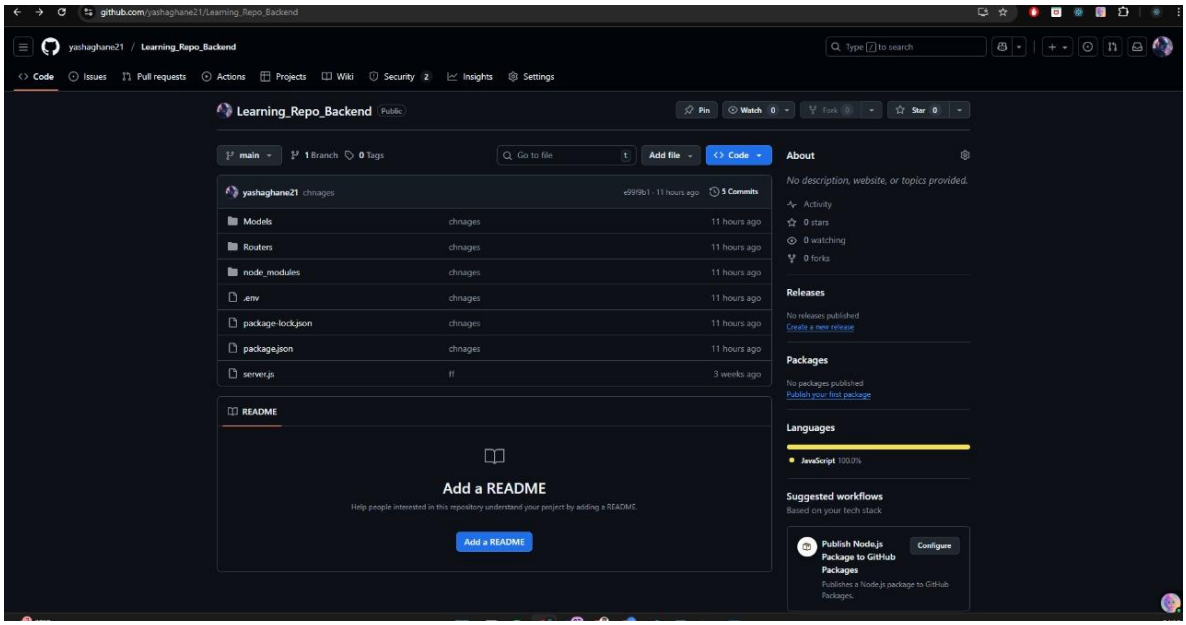
Procedure:

1. Initialize a repository in Git (e.g., git init or clone from GitHub).
2. Create an initial specification document or program and commit it.
3. Modify the specification to represent Version 1.1 and commit changes.
4. Repeat to create multiple versions (e.g., V1.2, V2.0).
5. Use branching to maintain experimental changes and merge them into the main version after testing.
6. Compare differences between versions using git diff or GUI tools.
7. Tag stable versions (e.g., v1.0, v2.0).

Output:

- Screenshots of repository initialization, commits, and version history (git log).
- Example of change specification file showing what was modified.
- Evidence of multiple versions stored in the SCM tool.





Applications:

- Tracks project evolution systematically.
- Enables collaborative development across teams.
- Provides rollback and recovery from errors.
- Supports release management and continuous integration.

Learning Outcomes:

- Understand the concept of Change Specification in software projects.
- Apply SCM tools to maintain multiple versions effectively.
- Develop skills in branching, merging, and version tracking.

Course Outcomes:

- Use SCM tools (e.g., Git) to manage project versions.
- Document change specifications and maintain systematic version history.

Conclusion:

Successfully learned how to make change specifications and apply an SCM tool (e.g., Git) to create and maintain multiple versions of software. This ensures traceability, collaboration, and better project management.

Theory Questions:

1. What is the importance of version control in software development?
2. How does change specification help in maintaining project history?

Case Study / Open-Ended Questions:

1. Suppose a team member introduces a bug in Version 2.0. How can SCM help you recover to a stable state?
2. How would you manage parallel development of two new features using branches in Git?

For Faculty Use

Correction Parameters	**Formative Assessment [40%]**	**Timely completion of Practical [40%]**	**Attendance / Learning Attitude [20%]**	
Marks Obtained				