# Experiment No. 10

**SEMESTER:** V (2024-2025) **DATE OF DECLARATION:** 07/10/24

**SUBJECT:** SE **DATE OF SUBMISSION:** 14/10/24

**NAME OF THE STUDENT:** Shaun Menezes **ROLL NO:** 40

| | |
|---|---|
| **AIM** | To use SCM tool for software configuration management for a case study "**Inefficiency in Timely Announcement of Holidays for Educational Institutions in Flood-Prone Areas**" |
| **LEARNING OBJECTIVE** | The student will apply software engineering methodology and principles to "**Inefficiency in Timely Announcement of Holidays for Educational Institutions in Flood-Prone Areas**" |
| **LEARNING OUTCOME** | The student will create the case study presentation for "**Inefficiency in Timely Announcement of Holidays for Educational Institutions in Flood-Prone Areas**" |
| **COURSE OUTCOME** | **CSL501.6:** Students will be able to implement and present a case study based on software engineering concept. |
| **PROGRAM OUTCOME** | **PO1:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.<br>**PO2**: **Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.<br>**PO3:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.<br>**PO4: Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.<br>**PO5: Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.<br>**PO8: Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.<br>**PO9:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings. |

| | |
|---|---|
| | **PO10: Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.<br>**PO11: Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments. |
| **BLOOM'S TAXONOMY LEVEL** | Understand<br>Apply<br>Analyze<br>Evaluate<br>Create |
| **THEORY** | 1. **Software Configuration Management (SCM)**<br>**Software Configuration Management (SCM)** is a discipline within software engineering concerned with systematically managing changes in software products throughout their lifecycle. SCM ensures consistency, integrity, and traceability of software by tracking changes made to the software's configuration items, such as source code, documentation, libraries, and other related resources. SCM plays a crucial role in facilitating coordination among team members, maintaining software quality, and preventing issues arising from conflicting changes.<br>2. **Key Concepts of SCM**<br>3. **Configuration Items (CIs)**:<br>o These are the components of the software system that are subject to configuration control. They can include source code, binaries, documentation, test cases, tools, and any other artifacts used in the development and maintenance process.<br>4. **Version Control**:<br>o Version control systems (VCS) are essential in SCM to track and manage changes to configuration items over time. It maintains a history of modifications, supports branching, and allows merging of changes made by different developers.<br>o Popular VCS tools include Git, Subversion (SVN), and Mercurial.<br>5. **Baseline**:<br>o A baseline is a formally reviewed and agreed-upon version of one or more configuration items, serving as a reference point for further development. It ensures that changes are only made to approved versions. |

6. **Change Management**:
    - Change management is the process of handling requests for modifications to the software configuration. It ensures that changes are properly evaluated, approved, and documented before implementation.
7. **Build Management**:
    - Build management involves creating an executable version of the software from the source code and other configuration items. SCM tools automate the process of compiling and assembling code, ensuring consistency in software builds across different environments.
8. **Release Management**:
    - SCM controls the process of releasing software to end users or production environments. This includes packaging, versioning, and distributing the software while ensuring that only approved versions are deployed.
9. **Audit and Review**:
    - SCM supports periodic reviews and audits to verify that software configuration management policies are being followed. This ensures compliance with standards and prevents unauthorized changes to the software.
10. **Uses of SCM**

1. **Tracking and Controlling Changes**:
    - SCM provides mechanisms to track changes to software over time. This helps in identifying who made a change, what was changed, when it was changed, and why.
2. **Team Collaboration**:
    - In large teams, multiple developers may work on the same software project. SCM allows them to collaborate efficiently by managing concurrent changes and resolving conflicts when different changes affect the same parts of the codebase.
3. **Reproducibility**:
    - SCM ensures that any version of the software can be reproduced using the recorded configuration information, including specific versions of dependencies, libraries, and tools.
4. **Risk Management**:
    - By maintaining a history of changes, SCM can help identify and mitigate risks associated with new features or bug fixes. If a change introduces a problem, it can be quickly reverted to a previous stable version.
5. **Quality Assurance**:
    - SCM enforces a controlled environment where all changes are reviewed and tested before being integrated into the main software build, improving overall software quality.
6. **Compliance and Auditing**:
    - In industries where regulations are strict (e.g., healthcare, finance), SCM helps ensure that software development and

maintenance processes meet compliance requirements by providing traceability and accountability.

11. **Functions of SCM**

1. **Identification**:
   - o SCM identifies and defines configuration items in a system and establishes a systematic control for versioning and managing these items.
2. **Control**:
   - o SCM controls changes to configuration items to prevent unauthorized changes and ensures that all changes are made according to an approved process.
3. **Status Accounting**:
   - o SCM records and reports on the status of configuration items throughout their lifecycle, such as versions, pending changes, and approved changes.
4. **Review and Audit**:
   - o SCM supports the review and auditing of configuration items to ensure that they conform to the required standards and specifications, and that the software development process adheres to established procedures.

12. **Benefits of SCM**

- **Improved Collaboration**: Enables teams to work together on large software projects without overwriting each other’s changes.
- **Version History**: Maintains a complete history of changes, allowing developers to revert to earlier versions of software when necessary.
- **Risk Reduction**: Provides a controlled environment for making and testing changes, reducing the likelihood of introducing errors.
- **Compliance**: Ensures that software development processes meet industry regulations by maintaining a clear and auditable history of changes.
- **Efficiency**: Automates many processes, such as builds and deployments, allowing for faster and more efficient software releases.

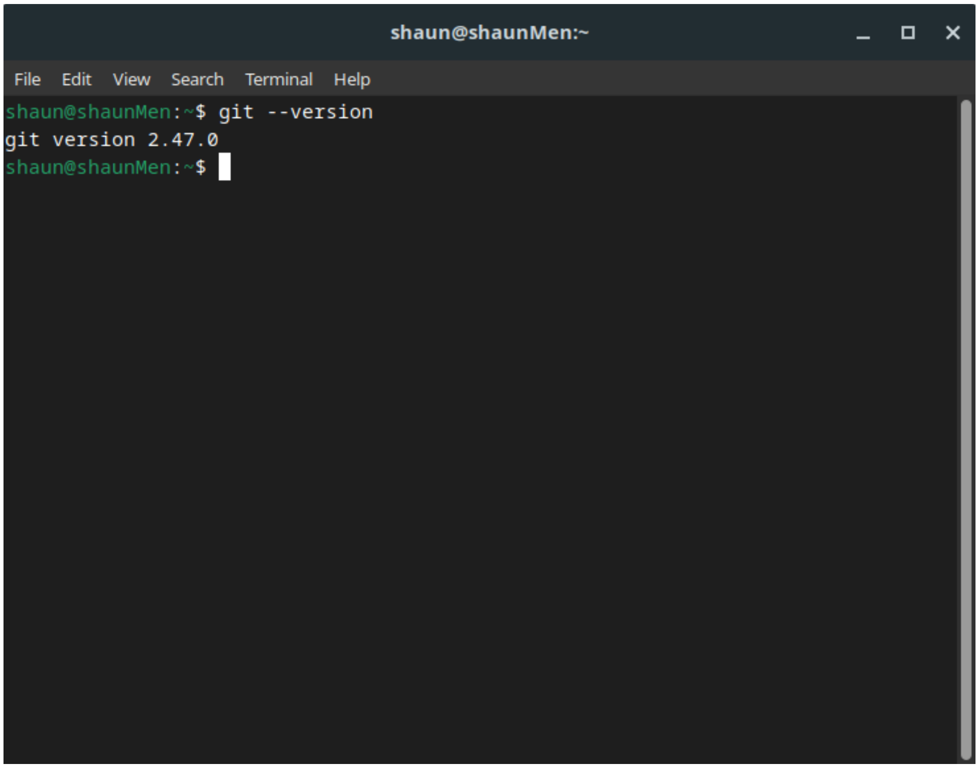
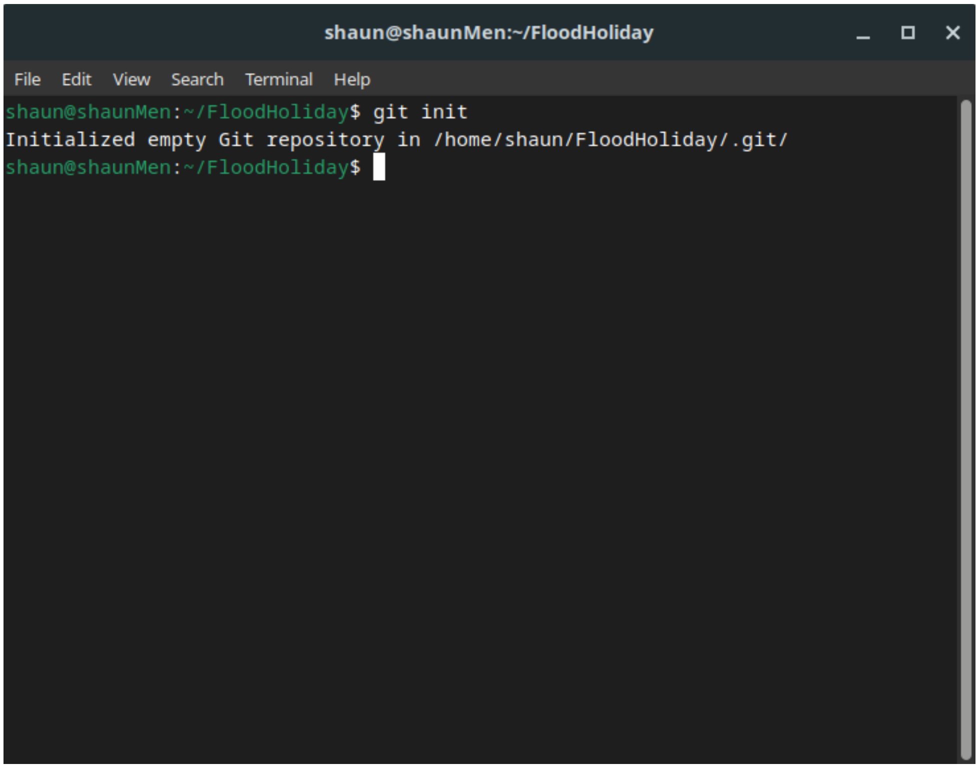
13. **Conclusion**

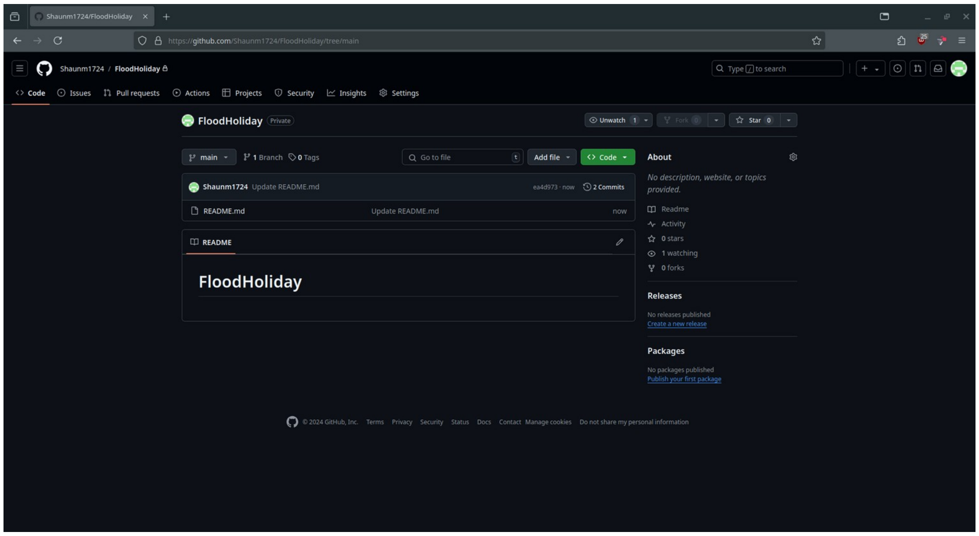
SCM is a vital aspect of modern software development that supports teams in managing complex projects by ensuring that changes are controlled, tracked, and implemented systematically. It helps prevent issues like version conflicts, undocumented changes, and deployment errors, leading to more reliable and maintainable software.

**LAB EXERCISE**

**SCM Tool: Git**

For this assignment, our group used Git as the Software Configuration Management (SCM) tool to manage the changes to our project code, documentation, and other configuration items. The tool helped us maintain version control, manage collaboration efficiently, and ensure a structured software development process.

| | |
|---|---|
| | <br> <br>`shaun@shaunMen:~$ git --version`<br>`git version 2.47.0`<br>`shaun@shaunMen:~$`<br>**Checking the Version of GIT installed**<br><br> shaun@shaunMen:~/FloodHoliday<br>File Edit View Search Terminal Help<br>`shaun@shaunMen:~/FloodHoliday$ git init`<br>`Initialized empty Git repository in /home/shaun/FloodHoliday/.git/`<br>`shaun@shaunMen:~/FloodHoliday$`<br>**Initializing an Empty GIT Repository** |

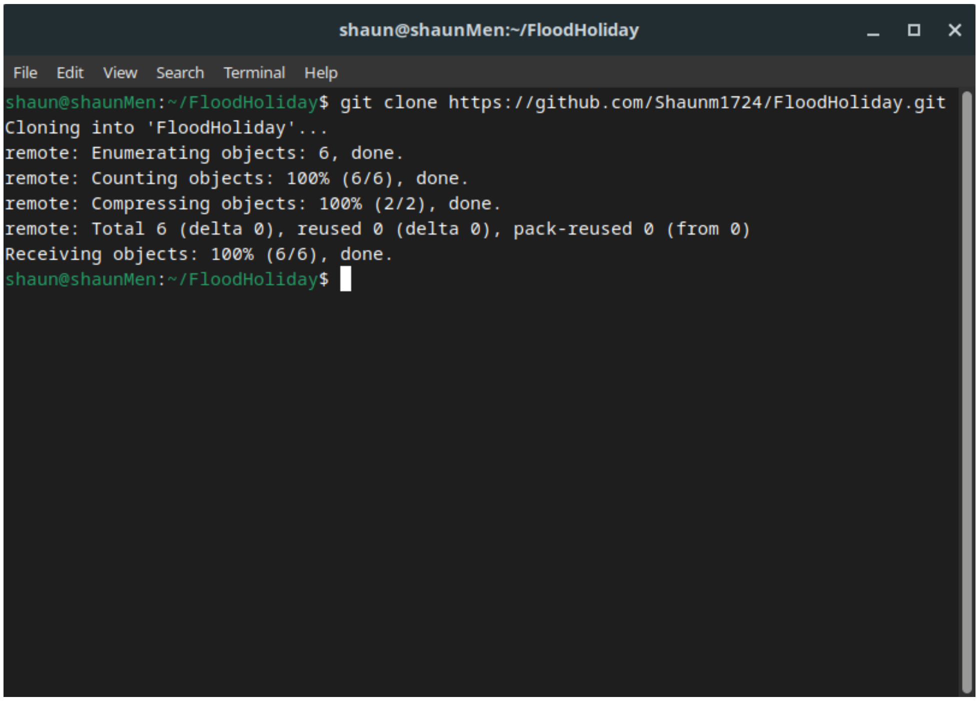


No description, website, or topics provided.

No releases published

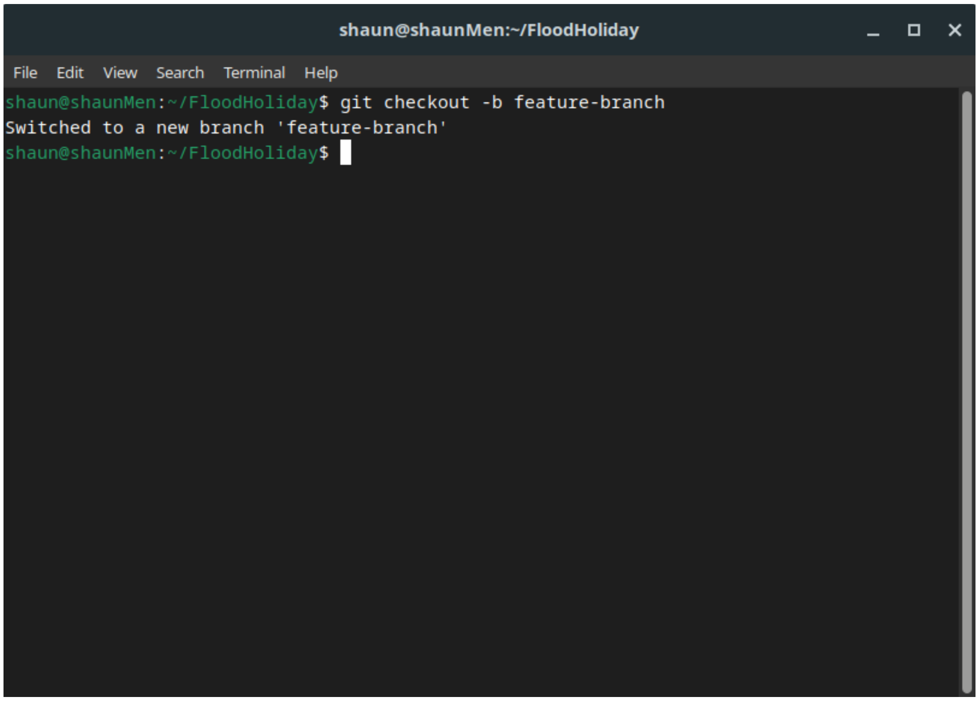
No packages published

**Creating of a Repository on GitHub Called FloodHoliday**



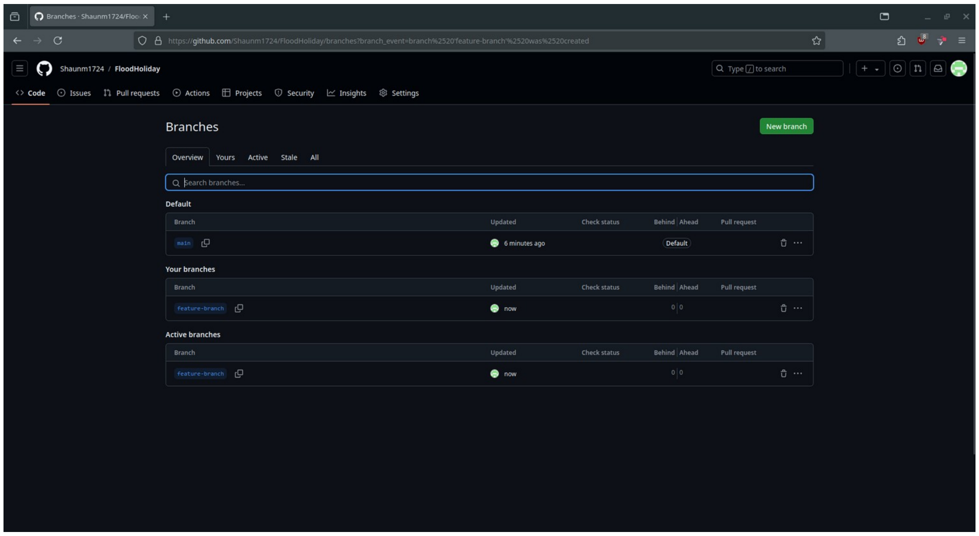
```
shaun@shaunMen:~/FloodHoliday$ git clone https://github.com/Shaunm1724/FloodHoliday.git
Cloning into 'FloodHoliday'...
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 6 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (6/6), done.
shaun@shaunMen:~/FloodHoliday$ 
```

**Cloning the GitHub Repository on another account**



```
shaun@shaunMen:~/FloodHoliday$ git checkout -b feature-branch
Switched to a new branch 'feature-branch'
shaun@shaunMen:~/FloodHoliday$
```

## Creating a New Branch for putting File to Avoid any Conflicts


Branches · Shaunm1724/Floo
https://github.com/Shaunm1724/FloodHoliday/branches?branch_event=branch%2520'feature-branch'%2520was%2520created
Shaunm1724 / FloodHoliday
Type / to search
Code
Issues
Pull requests
Actions
Projects
Security
Insights
Settings
New branch
Overview
Yours
Active
Stale
All
Search branches...

Branches

Default

| Branch | Updated | Check status | Behind \| Ahead | Pull request |
|---|---|---|---|---|
| main | 6 minutes ago | | Default | |

Your branches

| Branch | Updated | Check status | Behind \| Ahead | Pull request |
|---|---|---|---|---|
| feature-branch | now | | 0 \| 0 | |

Active branches

| Branch | Updated | Check status | Behind \| Ahead | Pull request |
|---|---|---|---|---|
| feature-branch | now | | 0 \| 0 | |

## Creation of the new Branch called “Feature-Branch”

**Conclusion**

The use of Git as an SCM tool greatly enhanced our group's workflow. We were able to collaborate efficiently, track changes, and maintain version control for our software project. GitHub served as an excellent platform for storing our remote repository and ensured that every member could access the latest version of the project.

The process we followed also reinforced the importance of software configuration management in real-world software development, ensuring that we could maintain software integrity and quality throughout the project lifecycle.

**Group Details**

| Sr. No. | Roll No. | Name of the Batch | Name of the Student |
|---|---|---|---|
| 1. | 34 | B | Aibal Biju |
| 2. | 35 | B | Ramya Kulkarni |
| 3. | 40 | B | Shaun Menezes |

**REFERENCES**

1. **https://www.atlassian.com/git/tutorials/what-is-version-control**
2. **https://www.ibm.com/docs/en/cm-sp/7.0?topic=management-software-configuration**
3. **https://git-scm.com/doc**