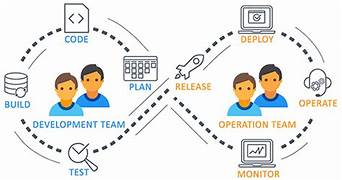
**Aim:** Practice Source code management on GitHub

**Description:**

GitHub helps the company’s long-standing efforts to accelerate development by breaking down communication barriers, shortening feedback loops, and automating tasks wherever possible.

***What is DevOps?***

DevOps is a software development and IT operations process that helps to improve the collaboration between developers and operators by automating the Deployment, configuration, monitoring, and management of applications.

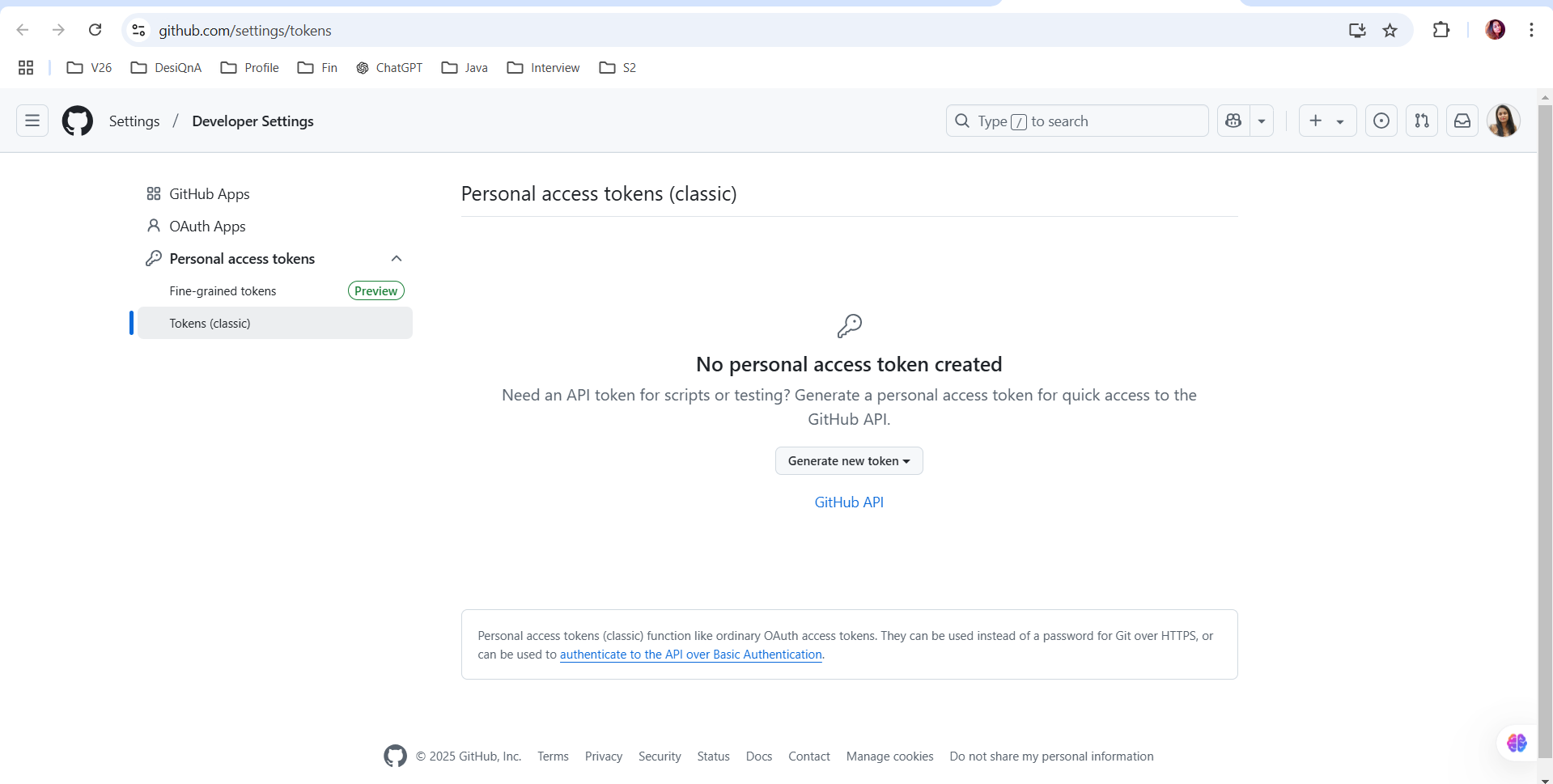


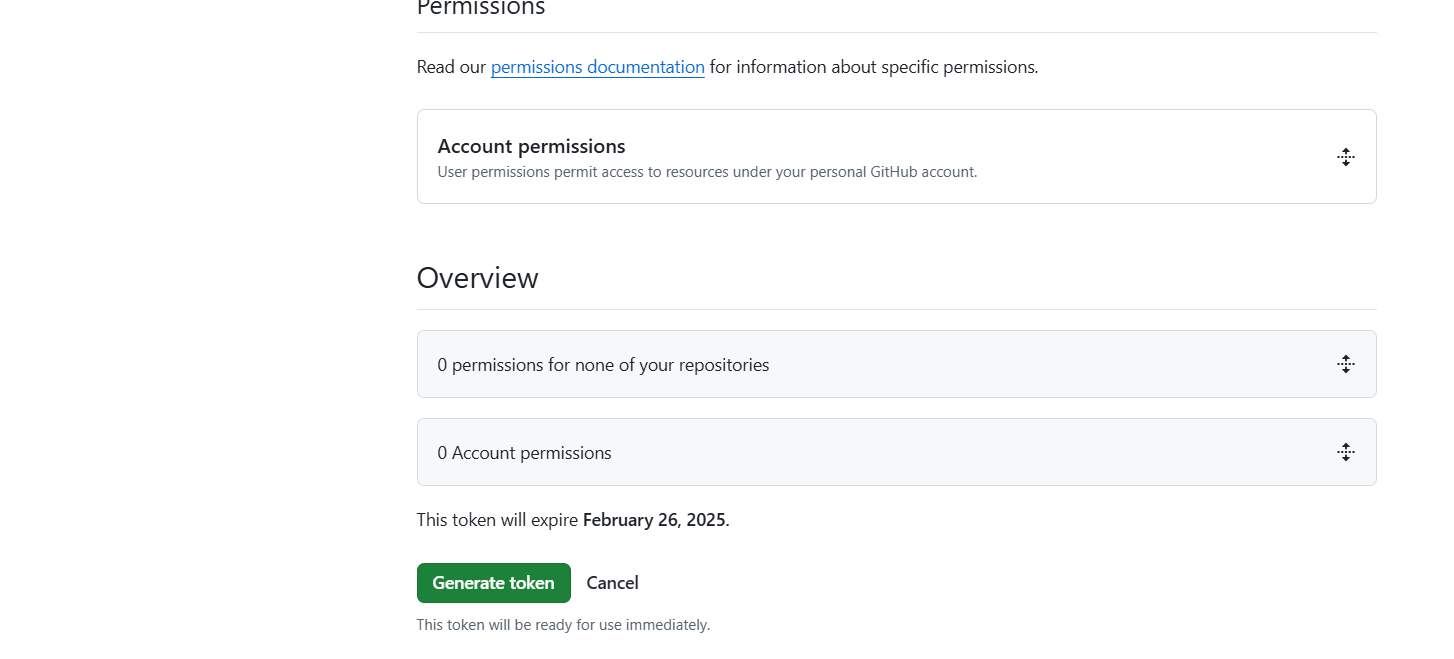
**Benefits of DevOps**

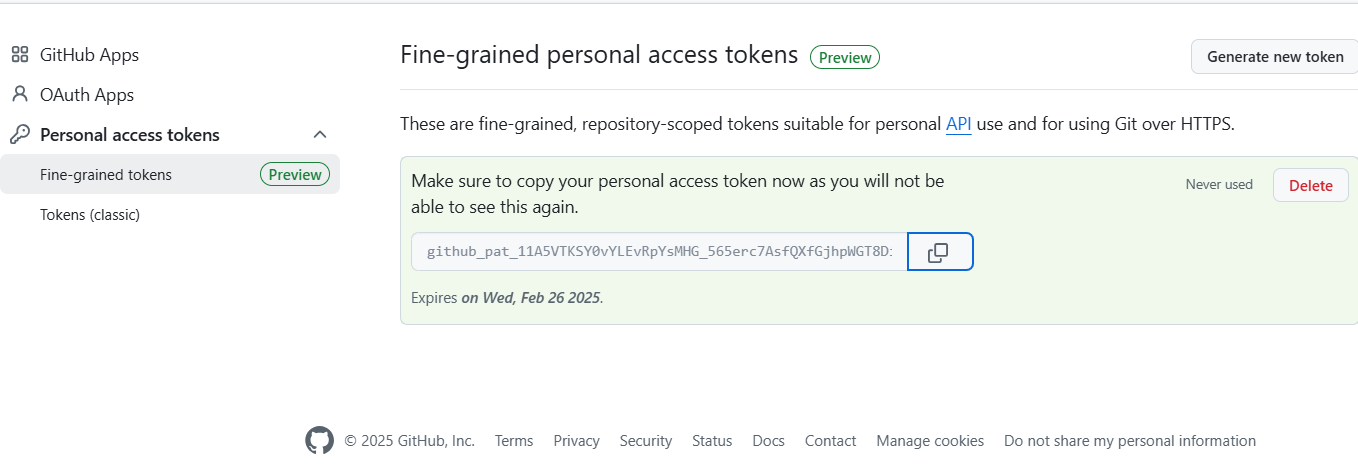
1. **Faster Delivery:**DevOps enables organizations to release new products and updates faster and more frequently, which can lead to a competitive advantage.
2. **Improved Collaboration:**DevOps promotes collaboration between development and operations teams, resulting in better communication, increased efficiency, and reduced friction.
3. **Improved Quality:**DevOps emphasizes automated testing and continuous integration, which helps to catch bugs early in the development process and improve the overall quality of software.
4. **Increased Automation:**DevOps enables organizations to automate many manual processes, freeing up time for more strategic work and reducing the risk of human error.
5. **Better Scalability:**DevOps enables organizations to quickly and efficiently scale their infrastructure to meet changing demands, improving the ability to respond to business needs.
6. **Increased Customer Satisfaction:**DevOps helps organizations to deliver new features and updates more quickly, which can result in increased customer satisfaction and loyalty.
7. **Improved Security:** DevOps promotes security best practices, such as continuous testing and monitoring, which can help to reduce the risk of security breaches and improve the overall security of an organization’s systems.
8. **Better Resource Utilization:**DevOps enables organizations to optimize their use of resources, including hardware, software, and personnel, which can result in cost savings and improved efficiency.

**Implementation:**

**Step 1: Sign-in in GitHub Account:**







github\_pat\_11A5VTKSY0vYLEvRpYsMHG\_565erc7AsfQXfGjhpWGT8Ditjvo5fNgA0CqhYRRkEAaSANQFQ5Ter3JGlLg

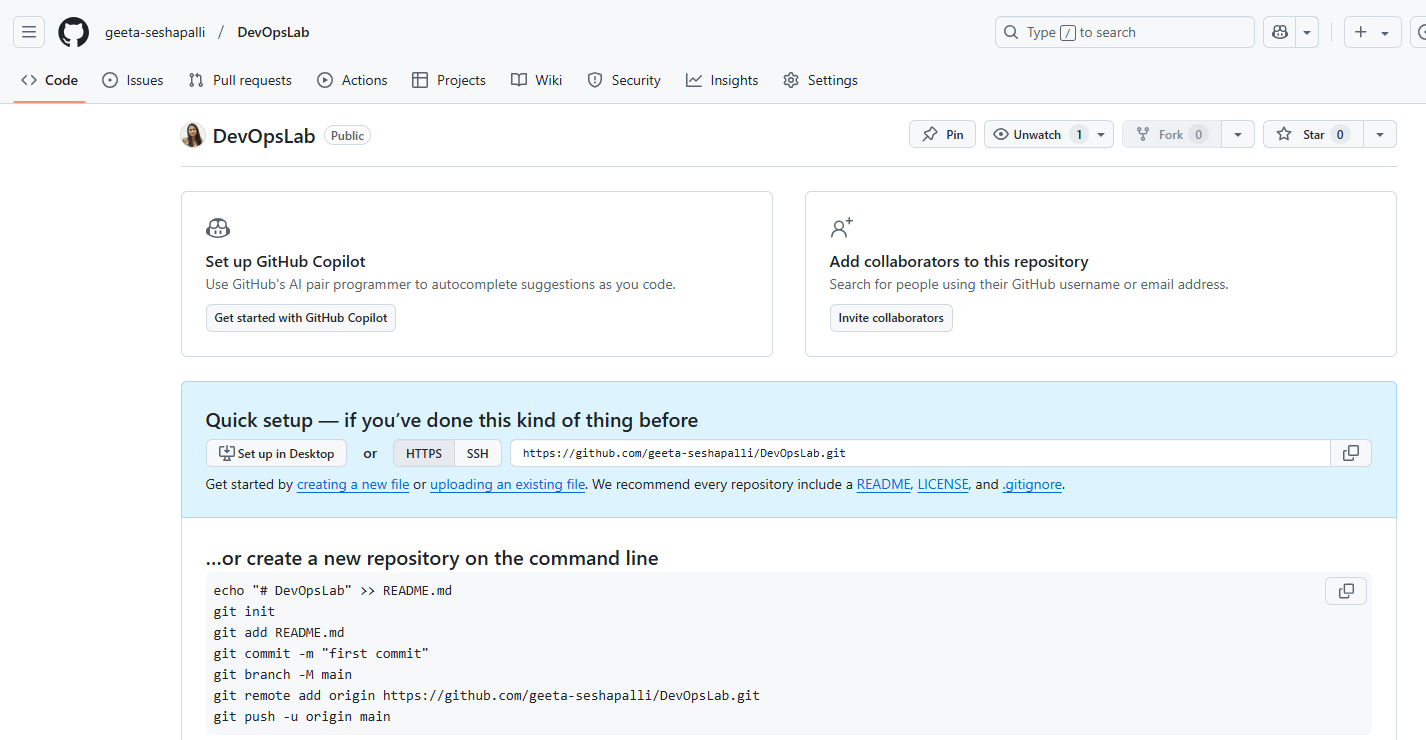
**Step 1A: Git Bash Sign-in**

***Commands:***

***git config --global user.name "Your Name"***

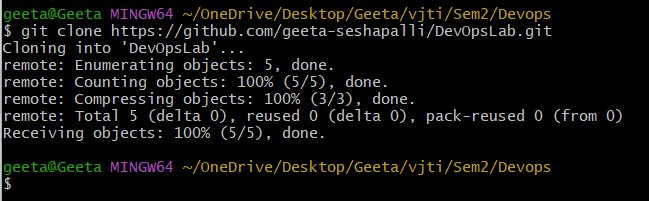
***git config --global user.email "your\_email@example.com"***





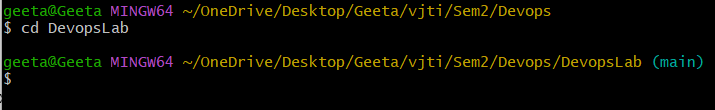
**Step 2: Clone the Repository to Your Local Machine**

***Command: git clone <repository-URL>***



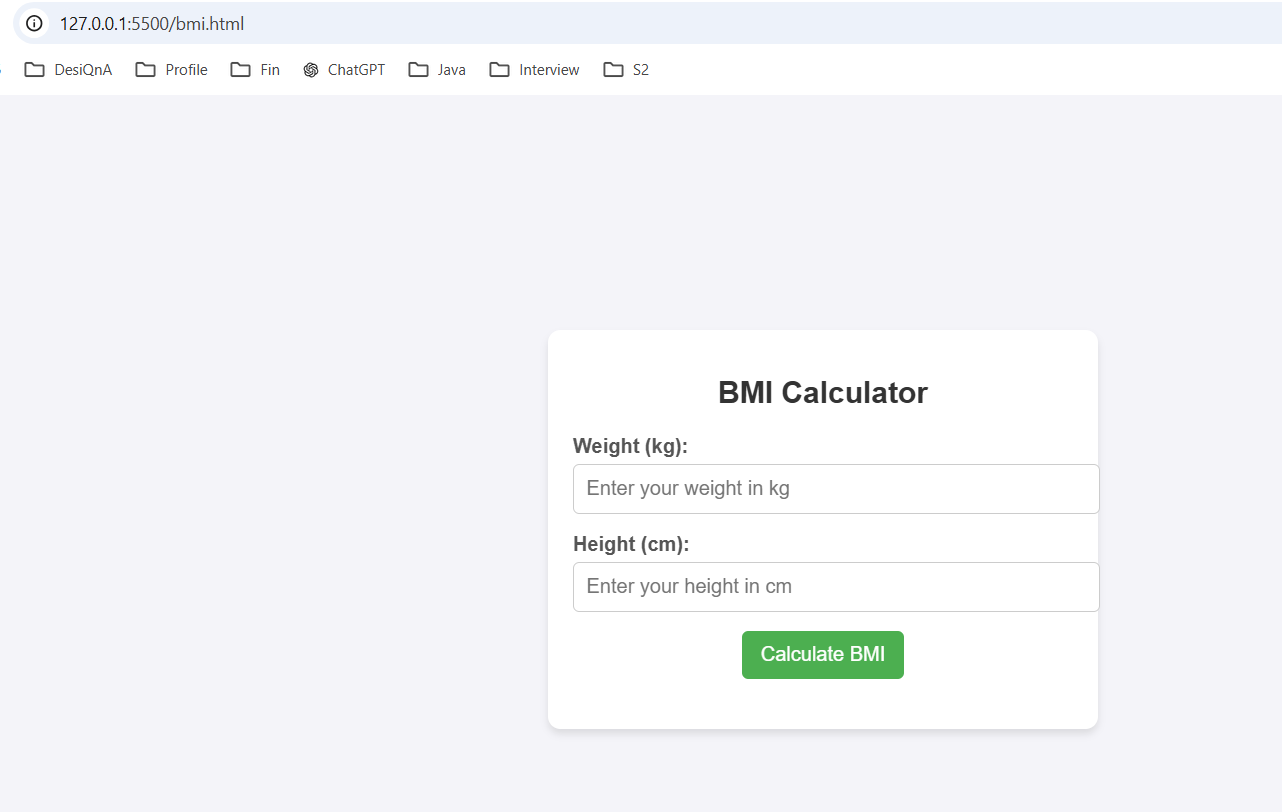
**Step 3: Move to the Repository Directory**

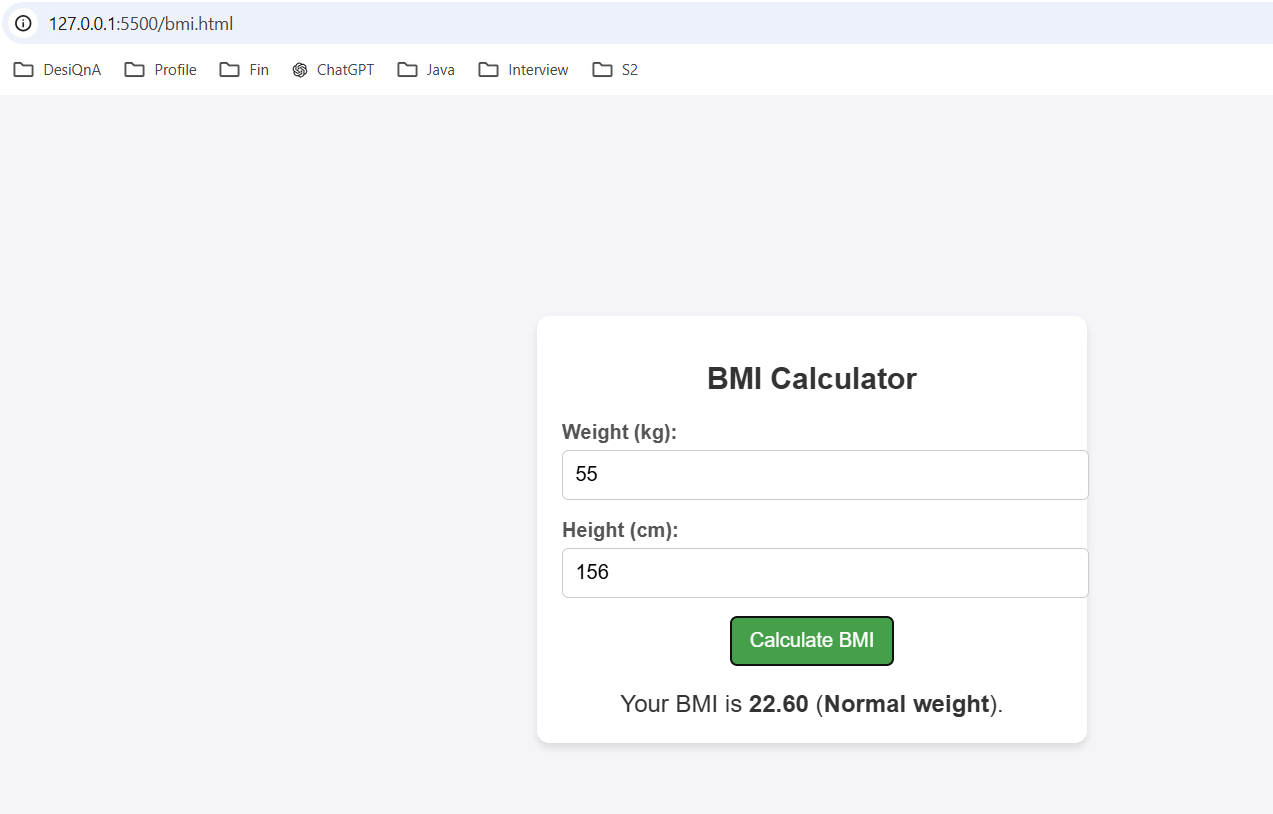
***Command: cd repository-name***



**Step 4: Create a New File in the Repository and Add the Source Code**

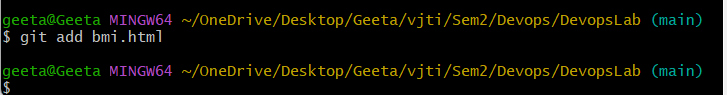
Created BMI Calculator



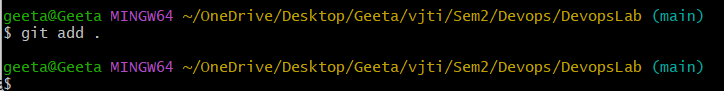


**Step 5: Stage the Changes**

***Command: git add filename.extension***



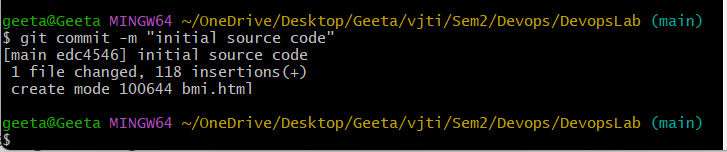
***To stage all changes, use: git add .***



**Step 6: Commit the Changes**

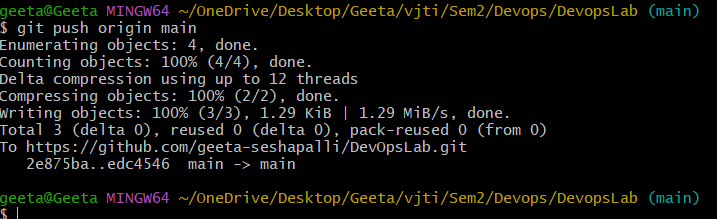
Commit the staged changes with a meaningful message:

***Command: git commit -m "initial source code"***

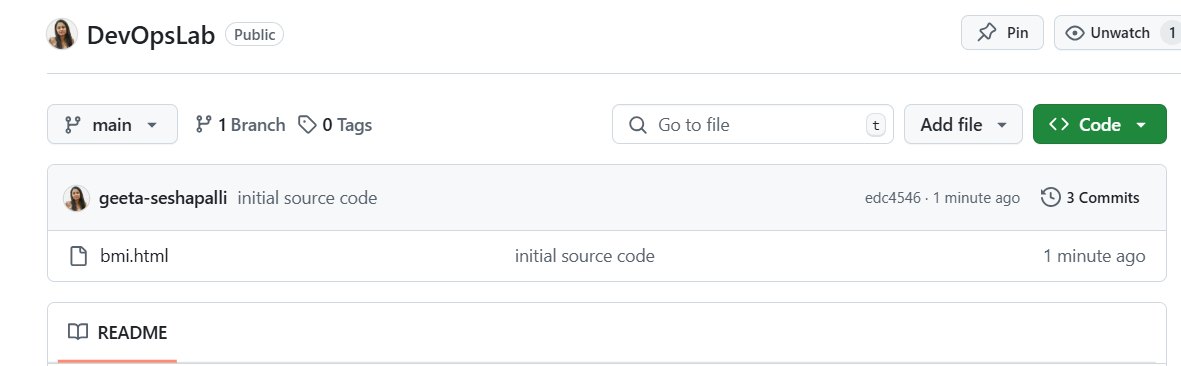


**Step 7: Push the Changes to the Remote Repository**

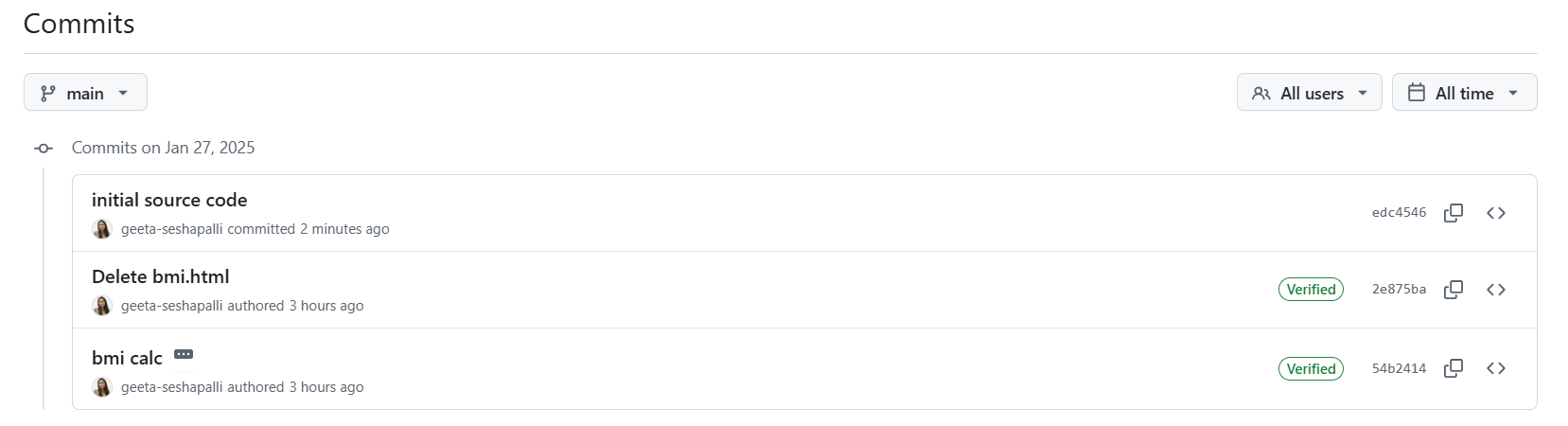
***Command: git push origin main***

******

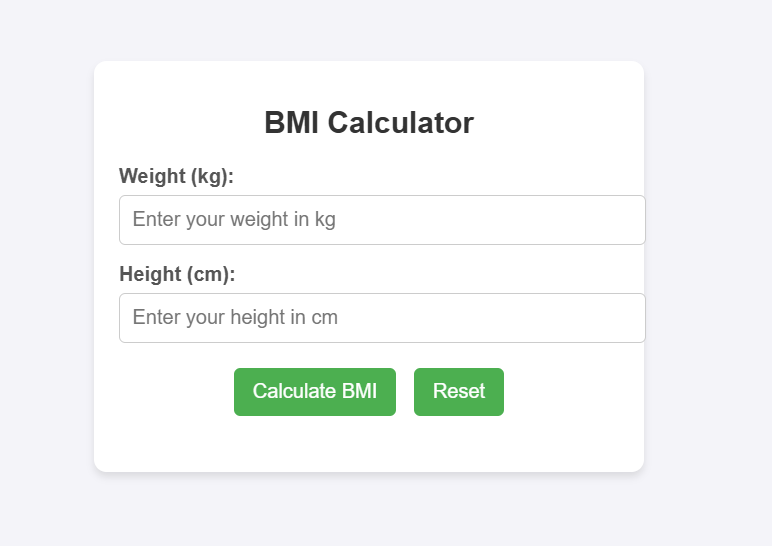
***Changes are invisible in GitHub Repository.***



***Commits are visible in GitHub Repository***



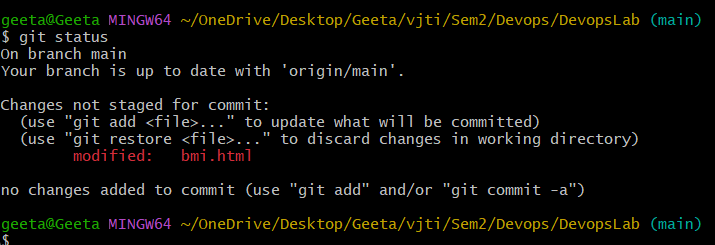
***New Changes made in existing code locally, added reset button***



**Step 8: Check the Status of Your Repository**

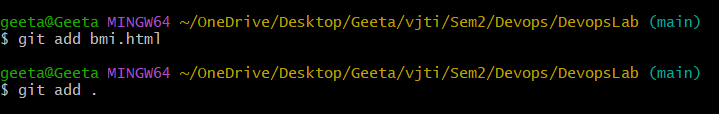
***Command: git status***

This will show a list of files that have been modified or are untracked.



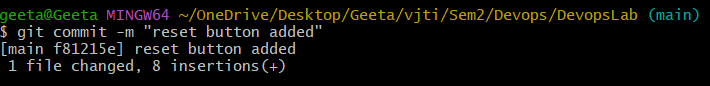
**Step 9: Stage the Changes**

***Command: git add filename.extension***



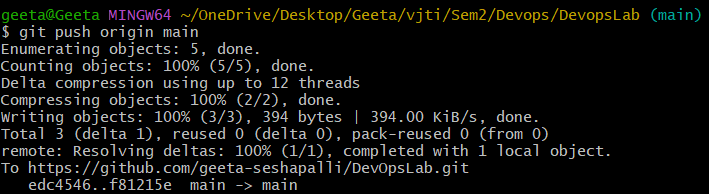
**Step 10: Commit the Changes**

***Command: git commit -m "Describe the changes made"***

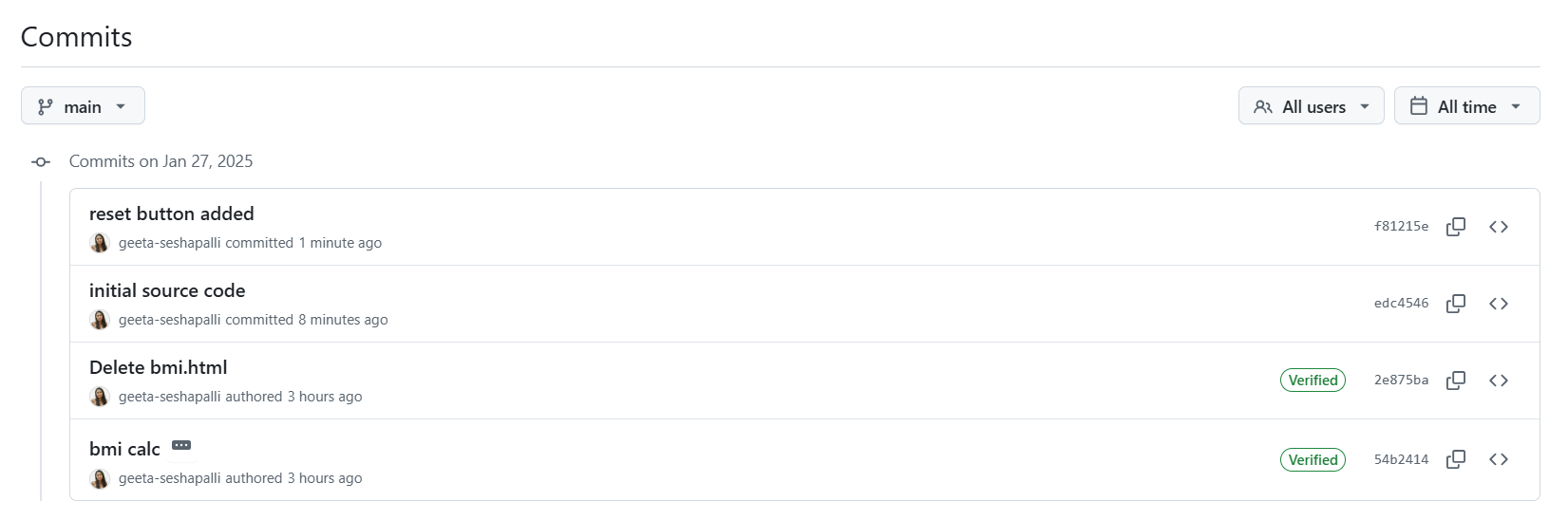


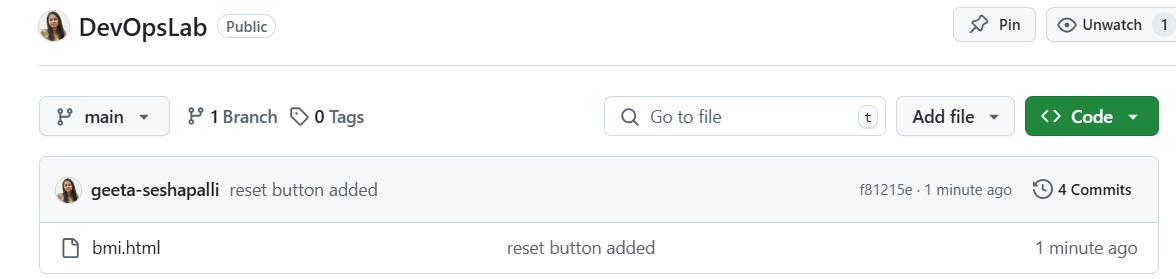
**Step 11: Push the Changes to GitHub**

***Command: git push origin main***



***Changes are visible in GitHub Repository***





**Conclusion:**

This experiment demonstrated the essential workflow for managing and versioning source code using Git and GitHub. Here's the summarized conclusion:

1. **Repository Management**:  
   Successfully created a new repository on GitHub and linked it to the local machine using Git.
2. **Version Control Workflow**:
   * **Cloning**: Retrieved the remote repository to the local system.
   * **Staging and Committing**: Staged and committed changes, ensuring a clear and trackable development process.
   * **Pushing Changes**: Synced local updates with the remote repository, making changes available to collaborators.
3. **Efficient Collaboration**:  
   Git and GitHub provide a seamless way to collaborate on projects, manage code versions, and maintain an organized history of changes.
4. **Practical Understanding**:  
   Gained hands-on experience with key Git commands (git clone, git add, git commit, git push), fostering a deeper understanding of version control best practices.

**Key Takeaways:**

* Git simplifies tracking changes in code and ensures no progress is lost.
* GitHub acts as a reliable platform for remote storage, collaboration, and backup.
* Writing meaningful commit messages improves project documentation and debugging.

**Repository Link:** [**https://github.com/geeta-seshapalli/DevOpsLab**](https://github.com/geeta-seshapalli/DevOpsLab)