Software Article

Implementing a Version Control Platform as an Interactive Educational Framework for Bioinformatics and Artificial Intelligence in Medicine

Alireza Ani[1,2], **Ahmad Vaez**[1,2,3]

[1]Department of Bioinformatics, Isfahan University of Medical Sciences, Isfahan, Iran, [2]Department of Epidemiology, University Medical Centre Groningen, University of Groningen, Groningen, The Netherlands, [3]Medical Image and Signal Processing Research Center, Isfahan University of Medical Sciences, Isfahan, Iran

Abstract

Background: Complex fields such as bioinformatics and artificial intelligence (AI) in medicine depend on big data and advanced computational tools. Consequently, they require structured platforms that enable collaborative development and academic capabilities. This report aims to demonstrate the pedagogical value of a version control system in an academic setting by deploying a dedicated local server for teaching AI in the field of medical sciences and developing bioinformatics software tools. **Methods:** We implemented a web-based, self-hosted platform for managing educational courses and developing AI-driven bioinformatics tools. This platform is based on the Git ecosystem as a decentralized version control system which facilitates collaboration among distributed teams. **Results:** The implemented platform enables comprehensive repository management and secure user access control, providing an integrated collaborative environment. It also supports educational activities, including course definition, sharing of teaching materials, and assignment submission. Within this framework, students and instructors can effectively coordinate tasks and store detailed project documentation throughout all stages of tool development. As a practical example, a new bioinformatics software package was successfully developed using this platform, demonstrating the system's effectiveness in supporting both educational and research-oriented software development. **Conclusion:** Our platform provides a secure and scalable environment for teaching and research in bioinformatics and AI in medicine. It provides advanced collaboration features that support learning activities and the development of computational tools, essential for improving both educational and research outcomes. We recommend its use in academic and research settings to strengthen the required skills and practices for advances in medicine and biology.

Keywords: *Artificial intelligence, computational biology, education, software design*

Submitted: 06-Jan-2026 **Revised:** 09-Feb-2026 **Accepted:** 16-Feb-2026 **Published:** 28-Aug-2026

Introduction

Bioinformatics is a complex and rapidly evolving field that inherently deals with big data, ranging from genomic sequences and protein structures to imaging information, and requires the use of diverse file formats and specialized analytical tools.[1] Moreover, each subdiscipline often depends on its own specific software. Such tools are typically developed by research teams worldwide, highlighting the ongoing need for platforms that support collaborative development and coordination.[2,3] Similarly, artificial intelligence (AI) in medicine has become indispensable both for medical practice and for educational purposes.[4,5] AI-driven tools enhance learning by enabling automation, interactive visualization, and adaptive instructions.[6]

Given the complexity and interdisciplinary nature of AI and biomedical sciences, education and tool development in these fields require a robust infrastructure that supports collaboration, reproducibility, and version control.[7] In the absence of such platforms, researchers and students often rely on informal or nonstandardized tools such as E-mails, instant messaging applications, social media, or general-purpose file-sharing services. While these tools offer convenience, they are not designed for scientific workflows and pose significant risks including reduced reproducibility and loss of version history.

To address this gap, implementing a version control system could provide a critical solution. Such a system enables

This is an open access article distributed under the terms of the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License (CC BY-NC-ND), where it is permissible to download and share the work provided it is properly cited. The work cannot be changed in any way or used commercially without permission from the journal.

For reprints contact: WKHLRPMedknow_reprints@wolterskluwer.com

How to cite this article: Ani A, Vaez A. Implementing a version control platform as an interactive educational framework for bioinformatics and artificial intelligence in medicine. J Med Sign Sens 2026;16:21.

Address for correspondence:
Dr. Ahmad Vaez, Department of Epidemiology, Unit of Genetic Epidemiology and Bioinformatics, University Medical Center Groningen, University of Groningen, Hanzeplein 1 (9713 GZ), P.O. Box 30.001, 9700 RB, Groningen, The Netherlands. E-mail: a.vaez@umcg.nl

Access this article online

Website: www.jmssjournal.net

DOI: 10.4103/jmss.jmss_2_26

Quick Response Code:

precise tracking of code evolution, facilitates effective collaboration among distributed teams, and ensures that every modification is documented and reversible.[8] Moreover, when integrated within an interactive educational framework, a version control platform not only promotes good scientific practice but also serves as a powerful educational tool, helping students and researchers to experience effective teamwork and to develop reproducible, collaborative, and transparent workflows in medical research.[9]

Version control systems have become fundamental tools in computational research and collaborative development environments in the recent years.[10,11] Among them, Git is the most widely adopted, primarily due to its flexibility, decentralized architecture, and open-source nature.[12,13] This technology was originally designed to support software development, and its distributed architecture allows each user to maintain a complete local copy of the repository, perform the assigned tasks, and subsequently integrate changes into the central application. Over the past decade, Git has evolved beyond its initial scope and now serves as a foundational tool for scientific collaboration and reproducible research.[14] Its primary strength lies in providing a structured framework for progress tracking and workflow transparency.[15] These features are essential in fields such as medical sciences and bioinformatics, where large datasets, iterative code development, and multi-institutional collaboration are common.

Despite its potential, integration of this technology into formal educational frameworks in medical science has received limited attention.[16] In educational settings, particularly AI in medicine and computational biology, Git offers a unique framework for engaging students and researchers in structured collaborative assignments.[8] Teaching AI and data science requires students to work on iterative, code-based projects that evolve rapidly. Hence, its principles of branching, merging, and version tracking mirror the iterative learning and research processes inherent in scientific discovery. Moreover, Git repositories allow for continuous integration and feedback, which can be crucial for effective supervision and learning. Instructors can track student progress, identify challenges, and provide precise feedback on specific code segments.[17] Git repositories not only store code but also serve as a backbone for documenting experiments, model updates, and data preprocessing workflows. Similarly, bioinformatics requires robust collaboration frameworks because it is a domain characterized by computational complexity and data-driven discovery. Therefore, Git helps maintain consistency across evolving analytical pipelines and facilitates integration across heterogeneous teams and institutions. In this project, we demonstrate how Git, when deployed on localized infrastructure, can be used as an effective and pedagogically valuable platform for education and tool development in bioinformatics and AI in medicine.

Materials and Methods

Server computer specifications

A dedicated local server environment was configured to host our version control system. The server was equipped with 2 terabytes of storage capacity, 64 gigabytes of random-access memory (RAM), and an Intel Xeon CPU architecture. Additional technical specifications included Ubuntu Server 22.04 LTS as the operating system and network configuration supporting both intranet and controlled external access. For data protection, weekly incremental backups were performed, and user access was restricted through firewall configuration. The server was managed using Apache 2, and secure shell was employed for administrative access, ensuring encrypted communication and secure remote management. This setup provided a reliable, secure, and reproducible environment for collaborative development and educational activities within the project.

Deployment of version control system

In this project, we selected Gitea (version 1.24.0)[18] as the version control platform due to its open-source licensing, active community support, and flexible deployment capabilities. To ensure full autonomy, strong data privacy, and independence from external service providers, Gitea enables institutions to host their own instances. This capability was particularly critical for our project and enabled us to operate within secure institutional infrastructures while maintaining compatibility with standard Git-based workflows. Additionally, Gitea provides a user-friendly web-based graphical interface, which simplifies Git procedures such as repository and user management. It also supports issue tracking, pull requests (PRs), and code reviews. Together, these tools assist users with limited command-line Git proficiency.

We utilized Gitea as a platform to facilitate structured educational activities focused on teaching bioinformatics and AI courses. Within the system, we created a dedicated repository to host all course materials. Sessions were organized as folders. User accounts were defined for students and instructors, enabling controlled access and collaborative engagement. Educational content including datasets, scripts, and instructional documents was uploaded to the repository. Figure 1 illustrates the overall workflow of the Git server, highlighting key processes and interactions.

Conducting a case study

As a case study, we used it for the development of a new R package. The platform enabled the team to define project objectives, track code versions, manage issues, and coordinate collaborative contributions efficiently. This demonstrates the versatility of Git platforms such as Gitea in supporting both skill development and practical scientific collaboration.

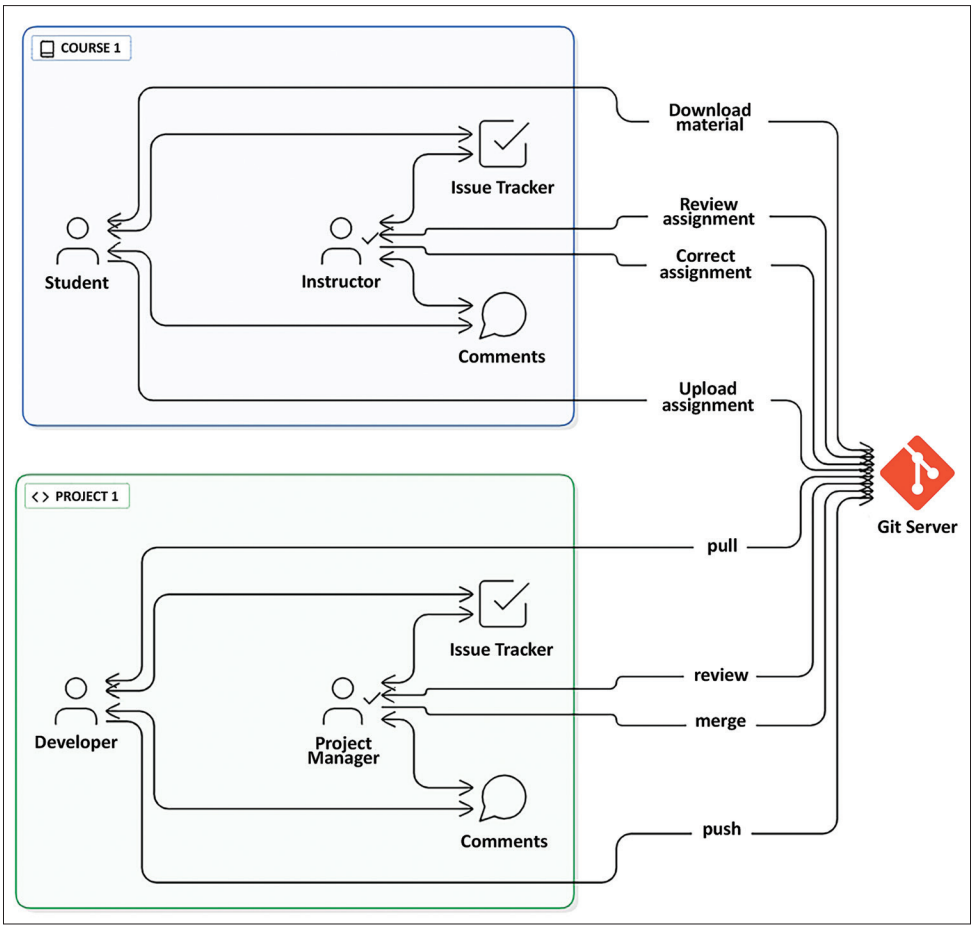


Figure 1: Git server workflow and version control process

Results

We established a version control system using the Gitea platform. The system was integrated into the student curriculum, serving as a supporting tool for teaching-related bioinformatics and AI in medicine courses. This infrastructure provides a unified, interactive environment for student–instructor communication, assignment submission, and learning. Instructors are able to monitor progress in real time and provide precise feedback via posted comments. Table 1 presents a comparative overview of the primary challenges encountered in bioinformatics and AI tool development and education, alongside the benefits of using a self-hosted Git system for these purposes. It outlines the key obstacles that educators and students may face, as well as the advantages this approach offers in these specialized fields.

Setting up repositories and user roles

The development process begins with the creation of a central repository on the Git server by the lead developer. This repository serves as the primary host for the software application's source code, which supports both collaborative development and educational use. The initial codebase, containing the first version of the software package, is transferred to the remote repository using standard Git functions ("add," "commit," and "push"). This process can be repeated for additional projects, and each is hosted in a separate repository.

User accounts were subsequently generated for developers and supervisors using the Git server's access control features. Developers were granted read/write permissions to enable code contributions, and supervisors were granted administrative privileges, including the ability to approve merges and manage issues. This role-based access control aligns with established practices in distributed version control systems and supports collaborative software engineering.

Student workflow: Forking, cloning, editing, and updating

Contributors or students first fork the central repository to their personal Git dashboards, in order to create an independent copy of the project. This method provides a safe environment for software development and exploring educational content without impacting the central repository. Developers can then clone their forked repositories locally to perform task-specific modifications, such as implementing features or resolving bugs.

Table 1: Comparative overview of current challenges and benefits of a self-hosted version control system for teaching bioinformatics and artificial intelligence in medicine

Current challenge	How Git helps	Drawbacks if not used
Fragmented and untracked code versions among students and supervisors	Centralized version control ensures traceability, version tracking, and reproducibility	Loss of code integrity, data redundancy, and irreproducible results
Limited collaboration and communication in team projects	Supports distributed development, issue tracking, and real-time supervision	Inefficient teamwork and lack of accountability
Lack of structured system for defining and managing course projects	Courses and repositories can be defined as structured projects with milestones and student groups	Uncoordinated course delivery and poor project organization
Educational materials scattered across multiple platforms	Shared repositories store lecture notes, datasets, and examples with version history	Outdated or inconsistent teaching materials
Manual or inefficient homework submission and grading	Students submit assignments via repositories; instructors review, comment, and grade via PRs	Increased administrative workload and reduced feedback quality
Limited or delayed feedback to students	Inline comments, issues, and PRs enable immediate, contextual feedback	Delayed feedback prevents timely corrections
Lack of practical training in professional workflows	Integrates Git-based learning directly into curriculum, teaching version control, documentation, and collaboration	Students miss essential skills for reproducible, team-based science
Poor integration between education and research development	Provides a unified, interactive platform for teaching, coding, and collaborative tool development	Disconnected learning experience and missed research-training synergy
Dependency on Internet-based commercial platforms	Local hosting allows optimal security and continued work during Internet outages	Security breach and project disruptions during network or service downtime

PRs: Pull requests

Upon completing their changes, developers push updates to their remote forked repositories. To propose integration into the central repository, they should submit PRs through the Git platform's interface. Each PR encapsulates a set of specific changes and facilitates structured code review. This workflow supports a modular and task-oriented development approach, allowing each component of the project to be built, tested, and refined independently. It also reinforces key educational objectives by encouraging students to maintain clear and well-documented commit histories. Ultimately, the workflow is designed to promote good software engineering habits and adherence to established best practices as intended learning outcomes.

Instructor workflow: Monitoring, review, feedback, and task management

Supervisors also fork and clone the central repository to maintain local control of the project. They can monitor ongoing developments by periodically pulling changes and reviewing developer contributions. Code reviews are conducted primarily via PRs, where supervisors examine code differences and provide inline feedback or suggest improvements. Approved contributions are merged into the main branch. In addition, supervisors manage project progress and task assignments through the platform's integrated tracking system. Overall, this approach enables efficient management of students and developers through structured task assignment and continuous progress monitoring.

Educational integration

In an educational perspective, Gitea can be incorporated into the curriculum in two complementary ways. First, it can be used for independent learning of Git technology. Students participate in a dedicated course on version control, learning Git fundamentals such as committing changes, branching, merging, and resolving conflicts. This training provides practical skills applicable to future team projects and collaborative workflows.

Second, it can be integrated with other courses. In this context, Gitea serves as a repository for educational materials, allowing students to explore shared resources, submit assignments, receive feedback, and complete exercises or homework directly within the platform. This approach promotes interactive learning and enhances communication between students and instructors.

A case study

The platform was also employed in the development of a bioinformatics R package, SNPannotator, from initial planning and coding to testing and documentation. The structured repository facilitated collaborative code development, feature integration, and version tracking in a team context. The resulting package provides a practical framework for efficiently deriving biologically meaningful insights from genome-wide association study data and for prioritizing candidate variants for functional validation. This software package is published on the CRAN repository[19] and has been used in multiple published studies.[20,21] A Shiny web interface was also developed for this package using the same platform.

Discussion

We implemented a self-hosted Git instance to provide an infrastructure that supports both educational functionalities and software development in bioinformatics and AI in medicine. The distributed architecture of the platform ensures robust version control and allows students to collaborate in real time. Each student can experiment in their own branch and review others' contributions, making the system an effective tool for teaching and learning software development through teamwork.[22]

Bioinformatics and AI in medicine are highly computational fields that rely on advanced tools and data-driven analysis. As these disciplines continue to evolve, the development of innovative and efficient software tools has become increasingly essential for new discoveries.[23] Achieving these goals requires sophisticated infrastructure to support team collaboration and software development from planning and coding to testing and deployment.[24] Version control systems effectively meet these needs by providing a structured framework for collaborative and reproducible software development across the entire lifecycle.[25] In addition, they can serve as educational platforms by providing teaching materials and facilitating mentorship.[26,27]

Git is currently the most popular version control system. Several tools such as GitHub,[28] GitLab,[29] Bitbucket,[30] and Gitea are built over this infrastructure, each offering specific features and deployment options. Gitea is a free open-source solution that combines a lightweight design with a user-friendly interface. It provides essential features for version control, collaboration, and project management.[31] These characteristics make it particularly well-suited for educational and research purposes. Importantly, Gitea repositories can be exported or migrated to other Git-based solutions such as GitLab if needed.

Therefore, for this study, we selected Gitea. This platform provides a powerful feature to track every change made to a project, including timestamps, comments, and authorship details. This enables users to easily trace the evolution of the code, revert to previous versions if necessary, and understand the rationale behind each modification. Without such version control, the historical context of changes and the evolution of the project would be lost.[32] Another key advantage of this approach is that it provides standardized methods for sharing, reviewing, and merging code across distributed teams. Git promotes transparency, consistency, and efficient collaboration through its robust version control and integration tools.[33]

It is important to highlight that integrating Git into the bioinformatics and AI curriculum will expose students to modern research workflows and collaborative tools used in professional environments. Through this experience, students will develop proficiency in fundamental skills that are essential for scientific research.[34] Moreover, previous reports indicate that incorporating Git into syllabus can significantly enhance student engagement and interaction with their studies.[35] Studies further highlight that both students and instructors express overall satisfaction with the inclusion of this technology in coursework. For instance, one study found that 75% of students were satisfied with using GitHub and requested its continued use in future semesters.[36] Another study reported that only 20% of students found the technology difficult to use, and around 80% stated that they would likely use it again in the future.[37] Although some students initially struggle to understand or adapt to the various features of Git platforms, these challenges are often resolved through dedicated tutorials early in the semester.[38] Likewise, instructors may experience difficulties adjusting to the new environment and its features and may benefit from additional support and training to ensure effective integration. The visual interface of the platform facilitates the learning curve for users with differing levels of technical proficiency.

It is also shown that using version control systems in university writing assessments provides a structured and transparent way to evaluate students' efforts. This approach breaks assignments into multiple milestones, with version control tracking each edit through timestamps and comments. This enables instructors to quantitatively assess writing progress over time and detect potential cheating or ghostwriting.[39]

Our experience demonstrates that Git-based supervision enhances mentorship quality. Instructors can monitor student progress, review commits, and provide contextual feedback in real time, which aligns with best practices in open science and reproducible research training. Hosting Git locally also ensured full control over repositories, which is aligned with institutional privacy policies. This setup reduces reliance on external platforms and eliminates the risks associated with service availability or data-sharing restrictions. More importantly, it enables uninterrupted scientific work during Internet outages.

As science becomes more data-driven and interdisciplinary, version control platforms, such as Git-based tools, can help coordinate work across computational biology, AI, and data science. They support the Findable, Accessible, Interoperable, and Reusable principles[40] and help ensure that scientific workflows remain transparent, reproducible, and easily shareable for collaborative research and education.[41] Our results suggest that Git-based technology should be encouraged in medical research institutions and graduate programs to enhance training and collaboration. Moreover, by demonstrating a practical implementation and highlighting the pedagogical value of a self-hosted Git server, this study provides both the rationale and platform for future sophisticated medical education research.

Conclusion

This study demonstrates that deploying a self-hosted Git instance provides a robust and secure framework for teaching bioinformatics and AI in medicine. By integrating Git into teaching and software development, we created a sustainable model that supports reproducibility, transparency, and collaborative learning. We recommend broader adoption of such platforms in bioinformatics and AI in medicine to enhance both research and educational outcomes.

Financial support and sponsorship

Nil.

Conflicts of interest

There are no conflicts of interest.

References

1. Ma XK, Yu Y, Huang T, Zhang D, Tian C, Tang W, *et al*. Bioinformatics software development: Principles and future directions. IL 2024;2:100083-11.
2. Kamoun C, Roméjon J, de Soyres H, Gallois A, Girard E, Hupé P. biogitflow: Development workflow protocols for bioinformatics pipelines with git and GitLab. F1000Res 2020;9:632.
3. Piccolo SR, Nathan A, Brazas MD, Kandpal M, Miró-Herrans AT, Kleinschmit AJ, *et al*. Opportunities and considerations for using artificial intelligence in bioinformatics education. Bioinform Adv 2025;5:vbaf169.
4. Ziemann M, Poulain P, Bora A. The five pillars of computational reproducibility: Bioinformatics and beyond. Brief Bioinform 2023;24:bbad375.
5. Ani A, Vaez A. Interdisciplinary research in Iran VII: The convergence of biology and artificial intelligence. J Med Signals Sens 2025;15:25.
6. Katona J, Gyonyoru KI. Integrating AI-based adaptive learning into the flipped classroom model to enhance engagement and learning outcomes. Comput Educ Artif Intell 2025;8:100392.
7. Khakpaki A. Advancements in artificial intelligence transforming medical education: A comprehensive overview. Med Educ Online 2025; 30: 2542807.
8. Chen KY, Toro-Moreno M, Subramaniam AR. GitHub enables collaborative and reproducible laboratory research. PLoS Biol 2025;23:e3003029.
9. Perkins M, Pregowska A. The role of artificial intelligence in higher medical education and the ethical challenges of its implementation. Artif Intell Health 2024;2:1-13.
10. Zolkifli NN, Ngah A, Deraman A. Version control system: A review. Procedia Comput Sci 2018;135:408-15.
11. Ram K. Git can facilitate greater reproducibility and increased transparency in science. Source Code Biol Med 2013;8:7.
12. Blischak JD, Davenport ER, Wilson G. A quick introduction to version control with Git and GitHub. PLoS Comput Biol 2016;12:e1004668.
13. Git. Available from: https://git-scm.com/. [Last accessed on 2025 Oct 20].
14. Russell PH, Johnson RL, Ananthan S, Harnke B, Carlson NE. A large-scale analysis of bioinformatics code on GitHub. PLoS One 2018;13:e0205898.
15. Perez-Riverol Y, Gatto L, Wang R, Sachsenberg T, Uszkoreit J, Leprevost Fda V, *et al*. Ten simple rules for taking advantage of Git and GitHub. PLoS Comput Biol 2016;12:e1004947.
16. Gallardo-Alba C, Grüning B, Serrano-Solano B. A constructivist-based proposal for bioinformatics teaching practices during lockdown. PLoS Comput Biol 2021;17:e1008922.
17. Chacon S, Straub B. Pro Git. 2nd ed. New York, NY: Apress; 2014. pp. 275-80.
18. Gitea Official Website. Available from: https://about.gitea.com/products/gitea/. [Last accessed on 2025 Oct 20].
19. Ani A, Kamali Z, Vaez A. SNPannotator: Investigating the Functional Characteristics of Selected SNPs and Their Vicinity Genomic Region 2023. Available from: https://cran.r-project.org/web/packages/SNPannotator/index.html. [Last accessed on 2025 Oct 20].
20. Lu X, van der Meer TP, Kamali Z, van Faassen M, Kema IP, van Beek AP, *et al*. A genome-wide association study of 24-hour urinary excretion of endocrine disrupting chemicals. Environ Int 2024;183:108396.
21. Rosenberg FM, van der Most PJ, Loman L, Kamali Z, Dittmar D, Snieder H, *et al*. A genome-wide association study of hand eczema identifies locus 20q13.33 and reveals genetic overlap with atopic dermatitis. Contact Dermatitis 2024;91:190-202.
22. Fehr J, Himpe C, Rave S, *et al*. Sustainable Research Software Hand-Over. Journal of Open Research Software 2021; 9: 1-10. Available from: https://openresearchsoftware.metajnl.com/articles/10.5334/jors. 307. [Last accessed on 2025 Oct 20].
23. Jamialahmadi H, Khalili-Tanha G, Nazari E, Rezaei-Tavirani M. Artificial intelligence and bioinformatics: A journey from traditional techniques to smart approaches. Gastroenterol Hepatol Bed Bench 2024;17:241-52.
24. Ferenc K, Rauluseviciute I, Hovan L, Kumar V, Kuijjer ML, Mathelier A. Improving bioinformatics software quality through teamwork. Bioinformatics 2024;40:btae632.
25. Braga PH, Hébert K, Hudgins EJ, Scott ER, Edwards BP, Sánchez Reyes LL, *et al*. Not just for programmers: How GitHub can accelerate collaborative and reproducible research in ecology and evolution. Methods Ecol Evol 2023;14:1364-80.
26. Using the Git Version Control System to Replace a Learning Management System. Available from: https://ieeexplore.ieee.org/document/10440614. [Last accessed on 2025 Oct 20].
27. Johnson G, Owen J. A survey of computer science academics relational to early introduction of version control systems. Procedia Comput Sci 2025;258:252-67.
28. GitHub 2025. GitHub Build and Ship Software on a Single, Collaborative Platform. Available from: https://github.com/. [Last accessed on 2025 Oct 21].
29. The Most-Comprehensive AI-powered DevSecOps platform. Available from: https://about.gitlab.com/. [Last accessed on 2025 Oct 21].
30. Atlassian. Bitbucket. Bitbucket | Git Solution for Teams Using Jira. Available from: https://bitbucket.org/product. [Last accessed on 2025 Oct 21].
31. Compared to Other Git Hosting | Gitea Documentation. Available from: https://docs.gitea.com/installation/comparison. [Last accessed on 2025 Oct 21].
32. Perkel JM. Challenge to scientists: Does your ten-year-old code still run? Nature 2020;584:656-8.
33. Abdill RJ, Talarico E, Grieneisen L. A how-to guide for code sharing in biology. PLoS Biol 2024;22:e3002815.
34. Alston JM, Rick JA. A beginner's guide to conducting reproducible research. Bull Ecol Soc Am 2021;102:e01801.
35. Gunnarsson S, Larsson P. Enhancing Student Engagement Using GitHub as an Educational Tool. Genombrottet, Lunds tekniska högskola. Available from: https://lucris.lub.lu.se/ws/

files/27854342/group1_github_final.pdf [Last accessed on 2025 Nov 28].
36. Nelson MA, Ponciano L. Experiences and insights from using Github Classroom to support Project-Based Courses. 2021 Third International Workshop on Software Engineering Education for the Next Generation (SEENG); 2021. p. 31-5.
37. Tiwari S, Rathore SS, Farooq SU, et al. How Students Are Using GitHub? A Survey. In: Proceedings of the 15th Innovations in Software Engineering Conference (ISEC 22). New York, NY: Association for Computing Machinery (ACM); 2022. p. 1–5.
38. Tushev M, Williams G, Mahmoud A. Using GitHub in large software engineering classes. An exploratory case study. Comput Sci Educ 2020;30:155-86.
39. Orbán LL. Using version control to document genuine effort in written assignments: a protocol with examples for universities. Front Educ. 2023;8:1169938. doi: 10.3389/feduc.2023.1169938.
40. Wilkinson MD, Dumontier M, Aalbersberg IJ, Appleton G, Axton M, Baak A, *et al*. The FAIR guiding principles for scientific data management and stewardship. Sci Data 2016;3:160018.
41. Weberpals J, Wang SV. The FAIRification of research in real-world evidence: A practical introduction to reproducible analytic workflows using Git and R. Pharmacoepidemiol Drug Saf 2024;33:e5740.