An Investigation Into the Perceived Effectiveness of GitHub Repositories to Teach Programming

Ebrahim Adam
School of Computer Science, The Independent Institute of Education, Varsity College, Durban, South Africa

eadam@varsitycollege.co.za

Abstract: The utilization of Git for version control and collaboration has long been the staple of the software development industry. To ensure that Computer Science students are industry-ready, academics adopt the use of platforms like GitHub in their teaching in different ways to expose students to these tools. In earlier studies, GitHub has been integrated for the purposes of student management, collection of assessments and facilitation of collaborative programming exercises. This paper explores the use of lecturer-provided GitHub repositories as a tool to support the teaching of core aspects of a programming curriculum. The pedagogical approach constructivism required students to learn core concepts by following repositories to build simpler applications. Thereafter, students are assigned more complex applications to develop. Students were required to work in pairs and submit their repositories for evaluation. This study adopted a case study approach supported by a combination of a student survey and a focus group discussion among lecturers and the teaching assistant involved in the course. Key findings included that integrating GitHub repositories presents several opportunities for collaboration, knowledge enhancement, self-directed learning, and reduced stress. However, this approach also resulted in team and technical challenges becoming prevalent, with students finding their own ways to overcome these challenges. Additionally, lecturers observed varied levels of engagement, reliance on lecturer-provided code, and occasional contextual challenges related to Internet connectivity. Lecturer-provided repositories can be enhanced in terms of structure, the inclusion of more comments explaining code, and through the inclusion of images and video. Recommendations are provided for academics who are looking to integrate GitHub into their teaching practice as well as suggestions for future work.

Keywords: Educational technology, Higher education, Computer science education, GitHub, Collaboration, Industry-readiness

1. Background

The field of software development is evolving rapidly, and it is becoming increasingly essential to equip Computer Science (CS) students with skills and attributes required in industry. These skills include the ability to work collaboratively in geographically distributed teams and utilize modern version control platforms like git to work on shared codebases with increasing complexity (Glassey, 2019, Patani et al., 2024). Upon completion of a CS qualification, graduates are expected to integrate into existing teams with existing and legacy codebases often hosted on git platforms, like GitHub (Patani et al., 2024). Knowledge of git also enables students to harness DevOps capabilities like automated testing, continuous integration, and continuous deployment (Patani et al., 2024). GitHub, a widely used cloud-based repository and version control platform with over 100 million users in 2023 (Dohmke, 2023), has been introduced several years ago into CS for managing assignments, assessments and providing feedback (Glassey, 2019).

This study aimed to investigate the perceived effectiveness of using GitHub repositories to teach final year programming from the perspective of students and lecturers. This paper provides a review of previous literature followed by into the course used as a context for this study. Then, an outline of the methodology is provided. Thereafter, an analysis of results is conducted in the context of existing literature. The paper concludes with recommendations and suggestions for future work.

2. Review of Previous Literature

This literature review commences by discussing the history of GitHub in higher education followed by discussing pedagogical considerations. Thereafter, strengths and challenges of using GitHub are as a tool are discussed. Within each sub-section, various themes are extracted from literature.

2.1 History of GitHub in Higher Education

Perhaps the earliest use of GitHub in higher education an American university in 2010 (Glassey, 2019). However, GitHub was not designed as an e-learning platform but was repurposed to facilitate learning (Glassey, 2019, Feliciano et al., 2016). In 2015, GitHub launched GitHub Classroom to allow for the management of individual and group submissions in repositories, automated grading of work, and collaboration with students (Gennarelli, 2018).

2.2 Pedagogical Considerations Relating to GitHub

GitHub offers benefits for teaching and learning, explained by Feliciano et al. (2016) as Contributing Student Pedagogy. In such a pedagogy, students are encouraged to participate in collaborative spaces to drive their own learning through equitable participation, creation of content, solution sharing, and participation in peer reviews (Feliciano et al., 2016). This approach aligns with the constructivist pedagogical approach adopted in teaching the course used in this study, whilst allowing connectivism to be harnessed (Siemens, 2007).

There are multiple ways in which GitHub can be integrated into a course. For managing assessments, students can submit assessments using GitHub, host their own projects, and GitHub Classroom spaces as a replacement to traditional LMS platforms to manage assessment submissions (Glassey, 2019, Feliciano et al., 2016, Haaranen and Lehtinen, 2015, Angulo and Aktunc, 2019). For feedback, lecturers can use the built-in issue tracker, pull requests to enable review, and unit testing to automatically grade work (Glassey, 2019, Hsing and Gennarelli, 2019). For disseminating course content, lecturers can share content, code or starter code with students (Haaranen and Lehtinen, 2015, Feliciano et al., 2016, Angulo and Aktunc, 2019, Patani et al., 2024). However, Patani et al. (2024) argues that these are passive approaches where students consume rather than create.

For fostering participation and collaboration, GitHub has been adopted for courses where students can participate by picking issues to fix, cloning, committing new changes, engaging in code reviews, and merging their changes (Zakiah and Fauzan, 2016). GitHub has also been used to facilitate practical lab activities (Angulo and Aktunc, 2019), for collaborative assessments where continuous integration (CI) was used to monitor code quality using DevOps tools (Lu et al., 2020, Raibulet and Arcelli Fontana, 2018).

2.3 Advantages Presented by GitHub

GitHub presents several possible advantages. Students are provided with skills to engage with industry tools, opportunities for increased collaboration, and a chance to contribute to their own learning (Glassey, 2019, Tushev et al., 2020). Academics are provided automated grading capabilities and insight into individual student participation in group projects (Buffardi, 2020, Feliciano et al., 2016, Angulo and Aktunc, 2019). Furthermore, academics may choose to reuse teaching material across years or courses (Feliciano et al., 2016). Institutions face reduced technical burden to setup and maintain dedicated infrastructure on campus for submissions of assignments (Glassey, 2019).

2.4 Challenges Confronting the use of GitHub

Despite the capabilities and advantages of GitHub, it is not a silver bullet (Tushev et al., 2020). There is a significant learning curve to understand the principles of git, learning git commands, dealing with merge conflicts, learning markdown, managing commits, and managing team dynamics around code collaboration (Glassey, 2019, Angulo and Aktunc, 2019, Tushev et al., 2020). Additionally, GitHub itself is also not the ideal space for different types of documentation documents and presentations (Tushev et al., 2020, Feliciano et al., 2016). Furthermore, academic integrity concerns become prevalent when students push their code to public repositories (Feliciano et al., 2016).

As can be gleaned from the literature, GitHub provides opportunities to enhance teaching and learning in CS, but it is not without challenges. Furthermore, whilst there have been several studies exploring GitHub for assessments and team collaboration, there has been limited study into how GitHub repositories have been used in developing context as a tool to support the teaching of core programming concepts through lecturer-provided repositories.

3. The Programming Course for This Study

The context for this study is a final year CS course at a private South African Higher Education Institution (HEI). The course covers design patterns and enterprise architecture. In the 2024 iteration, the course was co-taught by the author and two lecturers to two groups of students, respectively. The course was also supported by a postgraduate student who served as a teaching assistant.

The pedagogical approach undertaken in the course was grounded in constructivism and connectivism as explained by Siemens (2007). Previously, students were assigned smaller activities to facilitate learning of content. In this iteration, students were assigned a single complex application that students would develop over the semester. Students were required to work in pairs to build a web-based bookstore to a recommended architecture specification outlined in Figure 1.

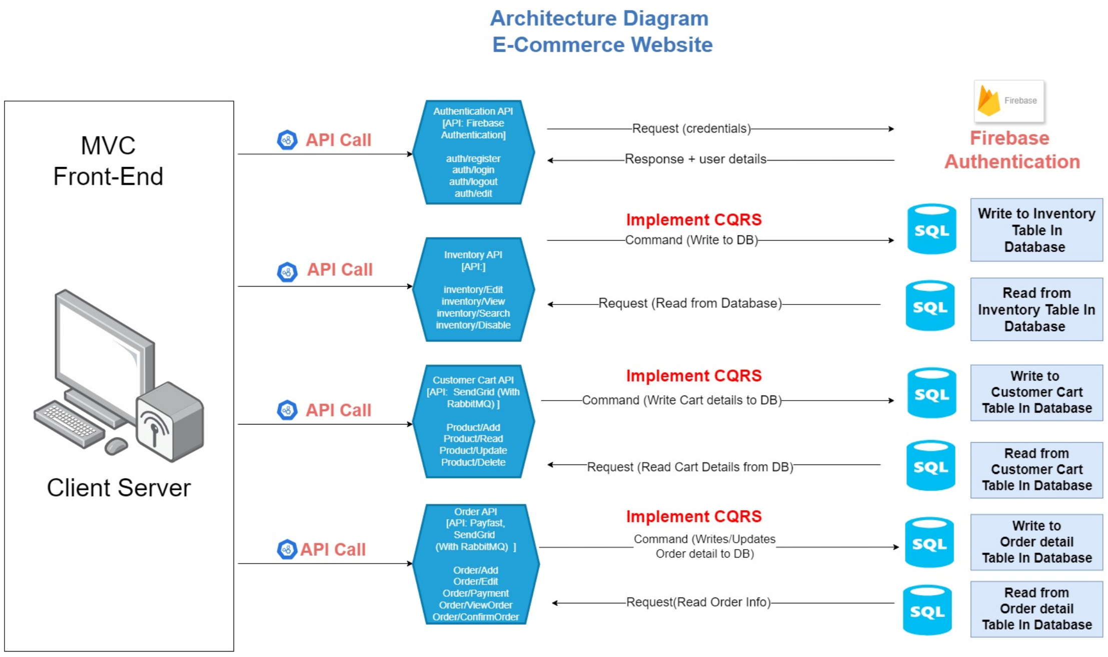
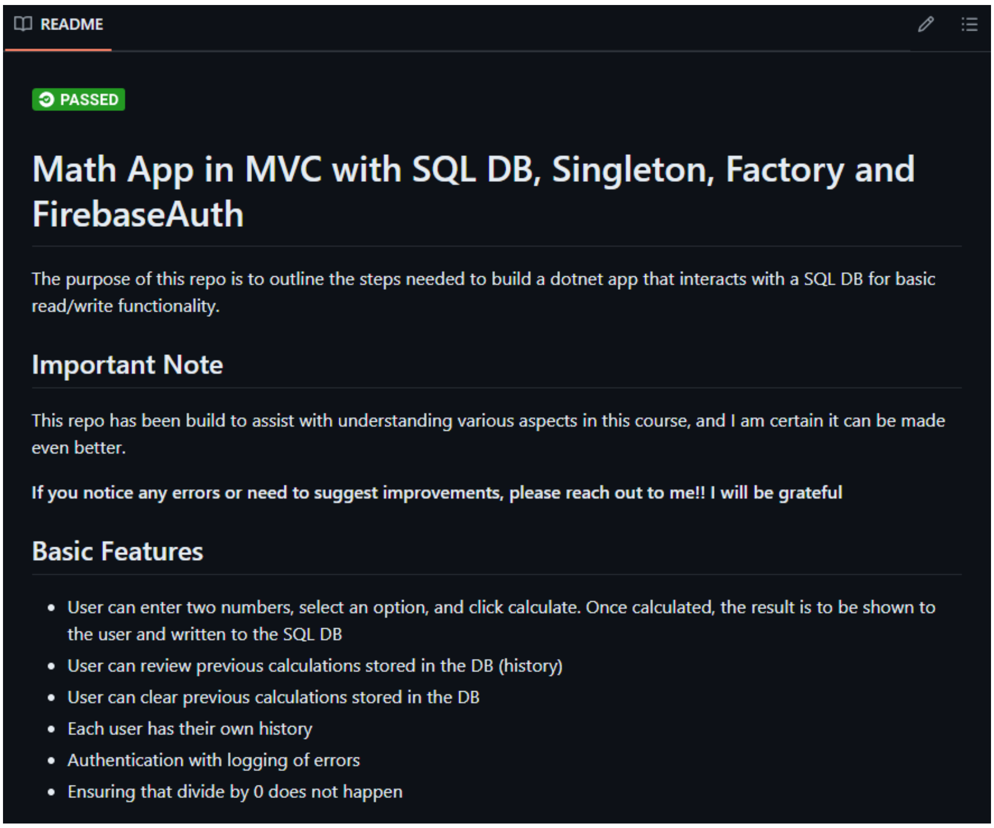


Figure 1: Online bookstore architecture

To teach the concepts, the author used a simple math application that incorporated an architecture of similar complexity developed over the semester using several GitHub repositories. Each repositories contained a readme file written in markdown with steps and explanations of course content in a tutorial format – see Figure 2. Working code for the simple math application was also pushed to the repositories as concepts were added. In this manner, students could follow the tutorial for their own learning and troubleshoot using the code.



README

PASSED

Math App in MVC with SQL DB, Singleton, Factory and FirebaseAuth

The purpose of this repo is to outline the steps needed to build a dotnet app that interacts with a SQL DB for basic read/write functionality.

Important Note

This repo has been build to assist with understanding various aspects in this course, and I am certain it can be made even better.

If you notice any errors or need to suggest improvements, please reach out to me!! I will be grateful

Basic Features

- User can enter two numbers, select an option, and click calculate. Once calculated, the result is to be shown to the user and written to the SQL DB
- User can review previous calculations stored in the DB (history)
- User can clear previous calculations stored in the DB
- Each user has their own history
- Authentication with logging of errors
- Ensuring that divide by 0 does not happen

Figure 2: An example of a tutorial from the repository showing instructions and badges

The various repositories and the concepts covered in each repository are outlined in Table 1. Repositories were hosted in the GitHub organization for the course: https://github.com/PROG7311-VCDN-2024/. As students were assigned an activity to integrate new concepts into their own bookstore, the lecturers used the repositories to teach the content.

Table 1: Repositories and concepts covered

Repository Name	Concepts Covered
MathApp	Revision of MVC in C#, FirebaseAuth in .NET, EntityFrameworkCore, integration of various design patterns, environment variables
MathAPI	Receiving and serving API requests, documenting with SwaggerUI
MathAPIClient	Making API requests, working with JSON objects, error handling
EmailService	Minimal APIs, SendGrid integration, GitHub Actions
MathApp (Docker branch)	Containerization using Docker and deployment to DockerHub
MathApp (CircleCI branch)	Continuous integration using CircleCI
MathApp (master branch)	Continuous deployment using CircleCI and Azure

4. Methodology

Since this study aimed to deeply understand a real-world phenomenon in a specific context without the intent of generalizing findings, an interpretivist paradigm was followed supported by a case study approach (Leedy and Ormrod, 2023). This study adopted a mixed methods approach to data collection to harness the benefits offered by both quantitative and qualitative instruments whilst allowing for triangulation to enhance the credibility of the findings (Harwell, 2011, Creswell, 2017). A case study approach has previously been adopted to understand how students experienced and perceived GitHub in several studies by Haaranen and Lehtinen (2015), Feliciano et al. (2016), Tushev et al. (2020), and Lu et al. (2020).

To collect data, this study employed an online survey distributed to students enrolled in the course as well as a focus group discussion with the two lecturers and teaching assistant. Data collection only commenced once institutional ethics was obtained (Reference: R.0002044), informed consent was obtained from all participants, and all participants' identities are protected and that participants are anonymised.

The link to complete the online survey was distributed to all 110 students enrolled in the PROG7311 course. These students comprised two thirds male and one third female, with an average age of 21 years old. Students were purposively sampled based on whether they actively engaged with the repositories provided. A total of 29 students provided complete responses resulting in a response rate of 26%. When considering the sampling approach, the fact that the survey was run outside of term when students were not on campus, and the fact that the survey was administered online, the response rate is deemed acceptable (Wu et al., 2022).

The practice of using a survey for a study exploring students use of GitHub has precedent in prior studies by Patani et al. (2024), Raibulet and Arcelli Fontana (2018), and Bennedsen et al. (2022). For this study, surveys that were used by Tushev et al. (2020) and (Haaranen and Lehtinen, 2015) to understand how students experienced GitHub to learn programming were adapted. Several open-ended questions were adapted from their surveys. The closed-ended Likert scale questions also adapted into this survey asked participants to provide their perceived GitHub competency before the course, their perceived GitHub competency after the course, and the extent to which lecturer-provided repositories enhanced learning. Responses were analysed using the Python pandas library to understand the descriptive statistics. Themes that emerged from the remaining open-ended survey questions were analysed using NVivo.

The focus group was hosted online to ensure that travel constraints did not hinder participation as the focus group was held outside of term (Sekaran and Bougie, 2016). The focus group was recorded and transcribed in Microsoft Teams. Thereafter, thematic analysis was also conducted using NVivo. The use of qualitative instruments to understand GitHub use for learning has precedent in a study Feliciano et al. (2016) who adopted a range of interviews to understand student experiences of using GitHub at various levels of study. In this study, the use of a focus group enabled participants and researcher to build upon responses provided by others in the focus group (Ravitch, 2020, Leedy and Ormrod, 2023).Analysis of results

This section commences with an analysis of the responses provided by students in the survey. Thereafter, an analysis of focus group discussion is provided. The themes that emerged from data included perceived benefits, perceived challenges, and suggestions for improvement. Within each of these themes, several sub-themes are explored relating to aspects like academic performance, problem solving, independent learning, student engagement, use of multimedia, and team collaboration. These formed the lens through which analysis is presented.

4.1 Students' Perceptions and Experiences of GitHub

As discussed in the literature, despite the necessity of learning GitHub, the technical learning curve was cited as an inhibiting factor to the adoption of GitHub (Tushev et al., 2020). One of the intentions behind adopting repositories as a central feature of this course was to encourage use of an industry tool among students (Bennedsen et al., 2022). Participants experienced an increased level of perceived competence with GitHub. Prior to engaging with the course, participants provided an overall positive rating (µ: 3.38, *M*: 3.0, *s*: 0.94). There was a noticeable increase reported after the course (µ: 4.1, *M*: 4.0, *s*: 1.01). Considering the std deviations before and after the course, it emerges that participants may have perceived varied levels of change in their perceived competence. In literature, students reporting improvements in GitHub competence was also found by Tushev et al. (2020), Patani et al. (2024) and Bennedsen et al. (2022). A distinguishing factor in this paper is the use of lecturer-provided repositories and participants indicated that the lecturer-provided repositories enhanced learning by providing an overall strongly positive rating (µ: 4.38, *M*: 5.0, *s*: 0.78). This suggests that their perceived competence was also shaped in a meaningful manner by the repositories.

4.1.1 Perceived benefits of using GitHub to learn programming

Several students reported that the course enhanced their knowledge by making them grow in their use of GitHub, "*Sprint tasks has allowed me to get completely comfortable with GitHub"* – SR3. Another student reported learning how to efficiently manage their repositories, *"It has made me do more research on GitHub practices and learn more about how to put manage my repo more efficiently"* – SR7. Similar findings were observed by Patani et al. (2024) whilst also showing that students experienced an increased sense of industry readiness through such tasks.

A further benefit was that students were able to hone their collaboration skills, *"Using GitHub provided more hands-on experience with collaborating with multiple users in the same repo"* - SR10. In alignment with this, another participant shared specific GitHub features that using repositories allowed them to develop, *"The sprints had helped me learn how to create branches, push and create pull requests"* – SR14. A similar sentiment was shared by another participant who indicated that these features enhanced collaboration, *"Using GitHub repositories to submit sprint work has greatly improved my understanding of version control, collaborative workflows, and project management. I've learned how to manage branches, create pull requests, and collaborate effectively with teammates through code reviews and issue tracking."* – SR20. The role of integrating GitHub to foster greater collaboration was also found by Patani et al. (2024). In developing collaboration skills, Feliciano et al. (2016) argue that students become more industry ready.

However, not all participants found that integrating repositories into the course benefitted them, primarily due to prior knowledge of GitHub, *"GitHub came as first nature I mean we are 3rd Computer Science students so being able to do something as simple as uploading proved no challenge"* – SR1. This was echoed by another participant as well, "*Not much because it's similar to other submissions"* – SR4. These participants bear resemblance to some surveyed by Bennedsen et al. (2022) who only saw the short term benefits of using GitHub.

4.1.2 Perceived challenges of using GitHub to learn programming

When considering challenges experienced by students, the most prevalent challenge was the learning curve. "*My teammates are not being as knowledgeable with using git."* – SR3. These findings relating to the learning curve and overcoming technical challenges were also observed by Feliciano et al. (2016), Patani et al. (2024) and Bennedsen et al. (2022). Students overcame these challenges by reaching out to lecturers, and this was cited by several participants. One example of such feedback was, *"I get advice and help from lecturers."* – SR13. Aside from lecturers there were also students turning to peers, *"I asked a friend who was more knowledgeable to help me."* – SR22. It also emerged that students undertook their own research when facing challenges. This included using online resources, *"I managed to resolve the issue using Stackoverflow."* – SR9. Other platforms were also consulted as outlined by a participant, *"I got assistance by Googling, research, YouTube videos and consulting lecturers."* – SR24. As was observed in this study, when students overcome the learning curve in their own ways, it creates a sense of accomplishment and confidence (Haaranen and Lehtinen, 2015).

Using GitHub also presented challenges relating to collaboration and merging, *"Occasional miscommunications with team members regarding pushing and pulling code from the repo"* – SR10. This was explained by another participant, *"The common issue is managing conflicts when multiple team members are working on the same files or branches simultaneously."* – SR20. To overcome this challenge, students adjusted their internal team processes to ensure that code collaboration was easier, *"My teammate and I had to let each other know when we made commits so we knew to pull before we started working."* – SR22. This was echoed by another participant, *"Resolving these conflicts requires communication and coordination among team members to ensure changes are integrated correctly."* – SR17. This proved useful for some, *"Better communication with fellow team members resolved challenges faced, and more hands-on practice made repo management easier"* – SR10. These challenges were also found by Feliciano et al. (2016).

4.1.3 Student suggestions for improvement

Participants were also asked to share how lecturer-provided repositories could be improved to enhance their learning. They indicated that documentation be provided in the forms of comments, *"More comments that even a non-technical guy can understand in the code"* – SR1. Another form of documentation that emerged was the need to support students with more steps coupled with guidance on possible errors and debugging, *"I think more walkthroughs and possible issues and errors that could occur would be more"* – SR7.

From a multimedia perspective, images were suggested to enhance engagement, *"Make them fun having images"* – SR1. Additionally, the inclusion of videos was cited as potentially beneficial, *"Perhaps a YouTube tutorial if necessary."* – SR15 and *"a video showing how the code works"* – SR16. Some participants highlighted topics which they felt would benefit from having videos, *"Include custom videos of how you did your activities, such as dockerizing."* – SR8 and *"videos may help to re-learn the git commands"* – SR13.

After experiencing GitHub repositories to learn at final year level, participants suggested that using GitHub within programming classes should be introduced in earlier years, *"I think if there was a short course in year one on how to push to git, big files, small files, etc. it would have opened us more to its usefulness."* SR11. This was echoed by another participant, *"The repos were good, you should start using them from year one. The sprint work too so that students aren't overwhelmed by the workload"* – SR16. Whilst there will always be room for improvement in how git is rolled out (Haaranen and Lehtinen, 2015), earlier integration was also argued for by Feliciano et al. (2016).

4.2 Lecturers' Perceptions and Experiences of Using GitHub to Teach

4.2.1 Perceived benefits of using GitHub to teach programming

When lecturers were asked about the perceived benefits of using GitHub to teach, they primarily identified benefits for students. The first benefit is an increase in academic performance and coding skills among students who engaged with repositories, *"When marking, I noticed thar students who used the repositories tended to perform better and put it more effort. There was a visible improvement in their practical coding skills"* – L3. Alongside this, lecturers observed an increase in students using other features of git, *"They've been come accustomed to it and to the use of markdown and use the other tools that are available from Git itself."* - L2. Tushev et al. (2020) also observed that students who engage begin to utilize other tools provided by GitHub.

Another benefit was that lecturer-provided repositories allow for self-paced learning, *"The repos were particularly useful for students who preferred to work at their own pace, providing a reliable resource for self-study."* – L1. This was echoed by another lecturer, *"There was benefit for students having a working codebase to refer to and this helped them learn through exploration and experimentation."* – L2.

Additionally, lecturers reported that students benefit from having what they perceived as a reliable resource, *"Having a reliable resource reduced students' stress, especially during assessments, as they had a clear reference to guide them."* – L1. Another lecturer shared a similar sentiment, *"Repositories provided a consistent and trustworthy resource, helping students feel more confident in their work."* – L3.

4.2.2 Perceived challenges of using GitHub to teach programming

Alongside the benefits, lecturers identified challenges relating to the use of GitHub, again from the view of their students. Lecturers found that not all students used the repositories effectively, *"While some students fully utilized the repositories, others did not engage as much, leading to inconsistencies in achieving learning outcomes."* – L1. Possible reasons cited were lack of motivated or struggling during the initial setup of GitHub, *"The repositories were not as effective for students who were not self-motivated or who struggled with the initial setup."* – L2. This aligns with a finding by Bennedsen et al. (2022) who found that initial setup proved challenging for some students.

One possible reason for a lack of motivation to engage was cited as students expecting that they would get the repositories which made them reliant on lecturer-provided code, *"Some students might rely too much on the provided code without fully understanding it, leading to superficial learning."* – L2. This practice resulted in students adopting an approach that negatively affected their engagement, *"Some students copied and pasted code without attempting to understand or modify it, which was more evident in less engaged students."* – L3.

In terms of technical issues, the learning curve proved significant for some students, *"Some students faced difficulties with the initial setup and needed additional support to get started with the repositories."* – L3. On occasion, lecturers found that students faced issues with the environment, *"Technical issues such as problems with virtual machines and internet connectivity that sometimes hindered the effective use of repositories."* – L2.

In this study, challenges from literature around repository privacy and the fact the GitHub was not designed as primarily a teaching tool did not emerge (Glassey, 2019, Angulo and Aktunc, 2019, Feliciano et al., 2016). The author posits that this is since repositories and student submissions in the course were hosted inside a GitHub organization linked to GitHub Classroom, which manages repository privacy and allows access to private repositories for lecturers.

4.2.3 Lecturer suggestions for improvement

Like was the case with students, lecturers were also asked to provide suggestions for improvement of the GitHub repositories to teach programming. Notable, there is alignment between students and lectures on what can be improved.

The first suggestion was that the codebase itself must not be provided to students before class, "*Perhaps a hybrid approach is needed, where initial exercises are done in class before providing complete solutions in the repositories, to ensure students first attempt the problems on their own."* – L2. This was supported by another participant to alleviate the reliance, *"I support this idea as it would help balance guided learning with independent practice."* – L1. Thus, whilst providing the repositories may be a benefit, there is a need for a more balanced approach.

Like with students, lecturers also suggested more instructions in the form of comments, *"I recommend providing more detailed instructions and comments within the repositories to help students understand the code better."* – L3. Additionally, lecturers also suggested more multimedia be included in repositories, *"Including visual aids and more interactive elements could further enhance the learning experience."* – L1.

Lecturers also suggested that repositories be phased in, *"Perhaps gradually introducing the use of repositories, especially for less experienced students, to ensure they first build a strong foundational understanding."* – L2. Considering this alongside students' feedback discuss earlier reinforces the need for integrating repositories into programming earlier on in the qualification would be beneficial.

5. Recommendations

Based on the findings of this study, the following recommendations, which are supported by literature, are made:

Integration is essential: It is recommended that lecturers teaching programming integrate the use of GitHub repositories into their teaching practice as findings show that expecting students to work GitHub for in-class activities encourages students to learn and master the technology. This prepares them well for a career in industry working in collaborative teams where git is widely used. Multimedia should be integrated into the repositories to increase engagement.

Overcome the learning curve: Students should be provided with training and resources on how to use GitHub to collaborate and submit their work. Lecturers should be upskilled on the advanced features of GitHub and on how to use repositories effectively and in a balanced manner within their teaching.

Integrate early on: It is also recommended that GitHub is integrated into teaching practice early on as findings suggest that students will have more time to traverse the learning curve presented by git as a technology. If this is done earlier, then students can adjust to using GitHub more frequently and can be exposed to more complex aspects like CI/CD later in the qualification. Since the code-collaboration tools provided by GitHub are where students reported challenges, group or pair activities are recommended even from early stages.

Integrate in moderation to encourage engagement: It is also recommended that lecturers do not provide the entire codebase up-front but may provide the tutorial-style structure to begin an activity. Once sufficient progress has been made by students, then the codebase may be shared with students to refer to. This will ensure that constructivism is applied, a balance is maintained, students engage with work and form their own knowledge, and are not reliant on lecturer-provided code. GitHub can be used to gather submissions and assess level of contribution.

6. Future Work

This study may be extended to different contexts, other courses and larger classes, to apply and evaluate the recommendations. Furthermore, the inclusion of more programming activities and DevOps tools into the learning process needs to be explored. Additionally, feedback and automated grading features of GitHub Classroom can benefit from further study. Lastly, the influence of integrating of GitHub earlier in a curriculum may be explored longitudinally.

References

Angulo, M. A. & Aktunc, O. 2019. Using GitHub as a teaching tool for programming courses. *2018 ASEE Gulf-Southwest Section Annual Meeting.* AT&T Executive Education and Conference Center, Austin, TX 78705: American Society for Engineering Education.

Bennedsen, J., Böjttjer, T. & Tola, D. Using GitHub classroom in teaching programming. 18th International CDIO Conference, 2022 Reykjavik University, Reykjavik, Iceland. pp. 690-702.

Buffardi, K. Assessing Individual Contributions to Software Engineering Projects with Git Logs and User Stories. Proceedings of the 51st ACM Technical Symposium on Computer Science Education, 2020 Portland, OR, USA. Association for Computing Machinery, pp. 650–656.

Creswell, J. W. 2017. *Research design: Qualitative, quantitative, and mixed methods approaches*, Sage publications.

Dohmke, T. 2023. *100 million developers and counting* [Online]. GitHub Inc. Available: https://github.blog/2023-01-25-100-million-developers-and-counting/ [Accessed 4 July 2024].

Feliciano, J., Storey, M. & Zagalsky, A. Student experiences using GitHub in software engineering courses: a case study. Proceedings of the 38th International Conference on Software Engineering Companion, 2016 Austin, Texas. Association for Computing Machinery, pp. 422–431.

Gennarelli, V. 2018. *Turning today's students into tomorrow's technologists with GitHub Education, a free program for schools* [Online]. GitHub Inc. Available: https://github.blog/2018-06-19-announcing-github-education/ [Accessed 5 July 2024].

Glassey, R. Adopting Git/Github within Teaching: A Survey of Tool Support. Proceedings of the ACM Conference on Global Computing Education, 2019 Chengdu, Sichuan, China. Association for Computing Machinery, pp. 143–149.

Haaranen, L. & Lehtinen, T. Teaching Git on the Side: Version Control System as a Course Platform. Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education, 2015 Vilnius, Lithuania. Association for Computing Machinery, pp. 87–92.

Harwell, M. R. 2011. Research Design in Qualitative/Quantitative/Mixed Methods. *The Sage Handbook for Research in Education: Pursuing ideas as the keystone of exemplary inquiry,* 2**,** pp. 147-164.

Hsing, C. & Gennarelli, V. Using GitHub in the Classroom Predicts Student Learning Outcomes and Classroom Experiences: Findings from a Survey of Students and Teachers. Proceedings of the 50th ACM Technical Symposium on Computer Science Education, 2019 Minneapolis, MN, USA. Association for Computing Machinery, pp. 672–678.

Leedy, P. D. & Ormrod, J. E. 2023. Practical Research: Planning and Design. 13th ed. Upper Saddle River: Pearson Upper Saddle River.

Lu, Y., Mao, X., Wang, T., Yin, G. & Li, Z. 2020. Improving students' programming quality with the continuous inspection process: a social coding perspective. *Frontiers of Computer Science,* 14, 5**,** pp. 1-18.

Patani, P., Tiwari, S. & Rathore, S. S. 2024. The impact of GitHub on students' learning and engagement in a software engineering course. *Computer Applications in Engineering Education***,** pp. e22775.

Raibulet, C. & Arcelli Fontana, F. 2018. Collaborative and teamwork software development in an undergraduate software engineering course. *Journal of Systems and Software,* 144**,** pp. 409-422.

Ravitch, S. M. 2020. *The Best Laid Plans… Qualitative Research Design During COVID-19* [Online]. Pennsylvania, USA: Sage. Available: https://www.socialsciencespace.com/2020/03/the-best-laid-plans-qualitative-research-design-during-covid-19/ [Accessed 7 July 2024].

Sekaran, U. & Bougie, R. 2016. *Research methods for business: A skill building approach*, John Wiley & Sons.

Siemens, G. 2007. Connectivism: Creating a learning ecology in distributed environments. *Didactics of microlearning. Concepts, discourses and examples***,** pp. 53-68.

Tushev, M., Williams, G. & Mahmoud, A. 2020. Using GitHub in large software engineering classes. An exploratory case study. *Computer Science Education,* 30, 2**,** pp. 155-186.

Wu, M. J., Zhao, K. & Fils Aime, F. 2022. Response rates of online surveys in published research: A meta-analysis. *Computers in Human Behavior Reports,* 7**,** pp. 100206.

Zakiah, A. & Fauzan, M. N. Collaborative Learning Model of Software Engineering using Github for informatics student. 2016 4th International Conference on Cyber and IT Service Management, 26-27 April 2016 2016 Bandung, Indonesia. IEEE, pp. 1-5.