FRONT-END DEVELOPMENT PLATFORM FOR ENHANCED LEARNING

THE IMPACT OF TUTORIAL-BASED LEARNING ON PROGRAMMING SKILLS

A Thesis Prospectus
In STS 4500
Presented to
The Faculty of the
School of Engineering and Applied Science
University of Virginia
In Partial Fulfillment of the Requirements for the Degree
Bachelor of Science in Computer Science

By
Mohammed Alwosaibi

November 8, 2024

Technical Team Members:
Mohammed Alwosaibi

ADVISORS

Ben Laugelli, Department of Engineering and Society

Yangfeng Ji, Department of Computer Science

Introduction

In today's world, programming skills are increasingly valuable across various fields, driving a demand among people from various backgrounds to learn how to code. However, many learners become trapped in an endless loop called tutorial hell, where they repeatedly engage in passive learning by watching tutorials, copying code, and answering programming-related questions without gaining practical experience (Kim & Ko, 2017). This learning method leaves developers underprepared in fields where transitioning from concepts to project-building is necessary. This ideology is mainly present in teaching front-end development, as the gap between understanding the concepts and how to apply them to create is vast. Addressing this sociotechnical challenge requires a tool that bridges the gap between passive learning and active skill-building, enabling learners to engage with actual projects.

I propose developing a web platform that provides users with practical front-end development projects to achieve this goal. The platform will display popular and complex websites (e.g., Netflix, Google) as templates of varying design difficulties. Users can choose a template and then use HTML, CSS, and JavaScript to replicate the design from scratch without external assistance. An AI model will assess their work, using image analysis to compare the user's design with the original, providing a score and constructive feedback. Since the challenge of tutorial hell has both technical and social roots, my STS research will examine a specific case of how integrating class and laboratory with hands-on exercises increased the number of students with higher grades (Handur et al., 2016). Using the Social Construction of Technology (SCOT) framework, I will analyze how learning tools shape programming practices and foster self-sufficiency among learners by examining the experiences of users who move beyond tutorials to independent coding.

This dual approach is essential for addressing the sociotechnical nature of the problem. If programming education focuses solely on technical improvements without addressing the social factors that hinder skill progression, learners may remain dependent on tutorials without building genuine, industry-relevant skills. Because the challenge of breaking free from tutorial hell is sociotechnical in nature, it requires attending to both its technical and social aspects to accomplish successfully. In what follows, I set out two related research proposals: a technical project proposal for developing a feedback-based interactive platform in order to apply knowledge and an STS project proposal for examining how hands-on learning played an important role in improving students' grades.

Technical Project Proposal

Currently, many coding platforms use tutorials that guide users step-by-step through coding exercises, promoting passive engagement. For instance, according to Kim and Ko (2017), most tutorials focus on basic concepts but fail to provide the learner with the problem-solving skills necessary for real-world applications. Some platforms now offer capstone projects with basic guidelines, where users have to come up with their own ideas instead of just copying examples. Another way people practice is by trying to recreate real websites. They usually do this on their own or by following YouTube videos that walk them through the steps. While this is a step toward independent work, it lacks an assessment component, which leaves a significant grade of uncertainty regarding the quality of one's work.

The primary limitation of these approaches is that they lack mechanisms for quality assessment and constructive feedback. For example, when learners replicate a website, there is no objective way to evaluate their work’s accuracy or effectiveness in fixed standards, such as responsiveness across devices, navigation functionality, mobile compatibility, or color consistency. As a result, even after extensive practice using these methodologies, learners may still be unable to create polished, professional websites.

According to Merrill, Reiser, Merrill, and Landes (1995), introducing a more feedback-driven platform could enable learners to gain hands-on experience and transition from passive to active learning. The proposed platform would allow users to select templates of well-known websites (e.g., Amazon, Google) with varying complexity, building them from scratch using the main front-end development languages. After completing their versions, an AI model would analyze the submissions and provide detailed feedback based on predetermined criteria. The platform’s structured feedback would address the significant gap left by existing approaches, where learners are uncertain about the quality and correctness of their work.

The proposed platform will offer a practical solution for learners in web development, emphasizing practice through real-world projects while providing critical feedback. Studies have shown the importance of feedback systems (usually through teaching assistants) and how they can massively improve one’s grasp of the material. Feedback systems, as noted by Anderson, Conrad, and Corbett (1989), play a crucial role in helping learners build a deeper understanding of concepts through iterative improvements. This approach helps users learn code basics and gets them thinking about design choices, user experience, and how to make sites work on different devices. With an AI-based assessment system, the platform will give users feedback on specific areas they can improve, covering details that tutorials or solo practice often miss. This approach encourages users to iteratively refine their skills, preparing them for the demands of professional front-end development.

The platform will use front-end development tools and AI for image analysis to provide users with detailed feedback. The AI will check user submissions against target designs, evaluating key aspects such as layout, responsiveness, and visual alignment to ensure adherence to professional standards. By leveraging advanced tools like TensorFlow or OpenCV, the platform will perform image analysis and pixel-by-pixel comparisons, offering users a clear understanding of how closely their designs match the original templates.

Studies on project-based learning, which indicate that projects improve skill acquisition and retention in programming, support the necessity and potential effectiveness of this platform. Studies show that learners who do project-based work perform better and remember more than those who only follow tutorials (Yeom, Herbert, & Ryu, 2022). By adding a structured feedback system, this platform supports proven educational methods, providing a better way to help learners retain information and build essential skills.

STS Project Proposal

In programming education, passive learning through tutorial-based instruction often leaves students without practical problem-solving skills. Tutorial-based instruction, where learners follow along with predefined steps without creating independent projects, contributes to the problem of tutorial hell (Malik & Zhu, 2023). This research focuses on the case of integrating hands-on programming exercises into educational practices, specifically examining how such exercises promote self-sufficiency in programming and reduce reliance on tutorials. The research question asks: how do hands-on exercises, when paired with structured assessments, influence students' independence in programming education?

This STS research examines a case study by Handur et al. (2016), which explores the integration of class and lab settings with hands-on exercises to improve students' understanding and performance in programming courses. The STS proposal will look at how integrating hands-on programming exercises reduces tutorial dependency and promotes self-sufficiency in programming education. Using the Social Construction of Technology (SCOT) framework, this research will analyze how the interaction between students, educational resources, and instructional methods influences the adoption of independent learning practices.

Some educational platforms, like FreeCodeCamp, incorporate capstone projects where learners apply concepts without explicit instructions. In addition, some studies examine the positive effects of replicating real-world projects, which can enhance practical skills, but they lack assessment tools to gauge the quality and accuracy of student work (Reese, 2011). Although these studies contribute to understanding how students learn programming, they rarely examine the social factors that shape tutorial dependency or provide insights into how learners move from passive learning to independent project creation. Additionally, research has focused on the tutorial as a technical tool rather than exploring how educational practices and social expectations affect the use of tutorials in learning. However, existing perspectives have not adequately addressed how structured feedback and project-based learning environments influence the transition from passive tutorial dependency to active skill-building. This gap in understanding forms the complication that this research seeks to address, as it reveals the limitations of current approaches in fully preparing students for independent work.

The limitations in existing approaches lie in their narrow focus on tutorials as a means of skill introduction, without accounting for the social norms that sustain tutorial dependency. Research has looked at how tutorials help beginners but often miss the challenges students face when trying to work independently. Studies on capstone projects also show that some students have trouble adapting without step-by-step guidance, pointing to a need for tools that build independent thinking and problem-solving skills (Handur et al., 2016). Additionally, while previous studies acknowledge that hands-on learning improves engagement and retention, they do not fully explore how these practices influence self-sufficiency. I describe self-sufficiency as the ability to feel satisfied with the work you create, which becomes more difficult without the help of step-by-step instructions. This gap in research calls for a closer look at the social and technical factors that shape students' programming practices.

This study gives readers a deeper understanding of the limits of tutorial-based programming by exploring why students often stay dependent on tutorials. If readers continue to adopt the traditional view that tutorials alone provide adequate programming education, they may overlook how these practices limit independent skill-building. Using SCOT, this research will reveal the social factors that contribute to students' reliance on tutorials and provide insights into how hands-on, project-based learning can better promote self-sufficiency.

This proposal argues that programming education needs to incorporate structured assessment and feedback mechanisms to help students transition from tutorial dependency to independent coding. This study’s main claim is that replicative, hands-on exercises can reduce tutorial dependency and improve self-sufficiency in programming (Mayer, 1981). This research will look at how students, educators, and resources interact to shape learning practices, showing why environments that encourage independent work and assessment are needed. These insights could help shift educational approaches toward more effective ways to train skilled programmers.

SCOT emphasizes the role of relevant social groups in shaping technological practices. In this case, students, educators, and programming resources are the primary groups influencing learning outcomes. According to Bijker, Hughes, and Pinch (1987), the process of technological development is socially constructed through iterations of a technology that address the priorities and concerns of various social groups involved in its design, a process shaped by concepts such as relevant social groups, interpretive flexibility, stabilization, and closure. SCOT's concept of interpretive flexibility is applied to examine how each group interprets and interacts with tutorials, projects, and assessment tools differently. This framework helps us better understand how social factors and teaching methods affect students' dependence on tutorials and their progress toward working independently in programming.

To support this, the research will look at data from Handur et al. (2016), which explores how combining classroom and lab work with hands-on projects impacts learning. Additional sources will include studies on the effectiveness of project-based learning (Xinogalos, 2016) and feedback systems in programming education (Van Merriënboer & Krammer, 1987). By reviewing these sources, this research will provide evidence of the impact of hands-on learning on students' programming skills and their ability to work independently.

Conclusion

To sum up, the technical project aims to build an interactive web platform to help users develop independent front-end skills. The platform will use AI-based image analysis to compare user-created website designs with templates, giving feedback on things like responsiveness and layout. This setup goes beyond traditional tutorials by offering practical, structured feedback that bridges the gap between passive learning and hands-on experience. It's designed to be particularly useful for beginner programmers and educators, addressing the limitations of tutorial-based formats by emphasizing real-world projects and independent skill development.

The STS project uses the SCOT framework to analyze how tutorial-based learning shapes programming habits and why students often remain dependent on such tools. By examining the interactions between students, educators, and programming resources, the research highlights how tutorial dependency is reinforced by both educational norms and technical practices. It also demonstrates the importance of addressing this dependency by incorporating structured feedback mechanisms and project-based exercises into programming education. This research provides new insights into the role of hands-on exercises and social factors in fostering independent learning, offering a deeper understanding of how these elements can reshape programming practices.

These insights directly inform the technical project by emphasizing the need to integrate assessment and feedback mechanisms into the platform. The SCOT framework reveals how learners' dependency on tutorials is shaped by common educational patterns, suggesting that a feedback-driven platform could reshape programming education. By covering both the technical and social sides of tutorial dependency, this prospectus shows how a balanced approach can help students become more self-sufficient in programming. This can make a real difference in helping learners move from beginners to skilled professionals.

Word Count: 2082

References

Handur, V., Kotecha, K., Narayanan, K., Parekh, P., & Rathod, H. (2016). Integrating class and laboratory with hands-on programming: Its benefits and challenges. *2016 IEEE 4th International Conference on MOOCs, Innovation and Technology in Education (MITE)*, 163–168. https://doi.org/10.1109/MITE.2016.041

Kim, A. S., & Ko, A. J. (2017). A pedagogical analysis of online coding tutorials. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (pp. 321–326). Association for Computing Machinery. https://doi.org/10.1145/3017680.3017728

Merrill, D. C., Reiser, B. J., Merrill, S. K., & Landes, S. (1995). Tutoring: Guided Learning by Doing. *Cognition and Instruction*, *13*(3), 315–372. https://doi.org/10.1207/s1532690xci1303_1

Anderson, J. R., Conrad, F. G., & Corbett, A. T. (1989). Skill acquisition and the LISP tutor. *Cognitive Science, 13*(4), 467–505. https://doi.org/10.1145/191033.191085

Yeom, S., Herbert, N., & Ryu, R. (2022). Project-based collaborative learning enhances students' programming performance. In *Proceedings of the 27th ACM Conference on Innovation and Technology in Computer Science Education Vol. 1* (pp. 248–254). Association for Computing Machinery. https://doi.org/10.1145/3502718.3524779

Bijker, W. E., Hughes, T. P., & Pinch, T. J. (Eds.). (1987). *The social construction of technological systems: New directions in the sociology and history of technology*. MIT Press.

Malik, K. M., & Zhu, M. (2023). Do project-based learning, hands-on activities, and flipped teaching enhance students' learning of introductory theoretical computing classes? *Education and Information Technologies, 28*, 3581–3604. https://doi.org/10.1007/s10639-022-11350-8

Reese, H. W. (2011). The learning-by-doing principle. *Behavioral Development Bulletin, 17*(1), 1–19. https://doi.org/10.1037/h0100597

Mayer, R. E. (1981). The psychology of how novices learn computer programming. *ACM Computing Surveys, 13*(1), 121–141. https://doi.org/10.1145/356835.356841

Xinogalos, S. (2016). Designing and deploying programming courses: Strategies, tools, difficulties, and pedagogy. *Education and Information Technologies, 21*, 559–588. https://doi.org/10.1007/s10639-014-9341-9

Van Merriënboer, J. J. G., & Krammer, H. P. M. (1987). Instructional strategies and tactics for the design of introductory computer programming courses in high school. *Instructional Science, 16*, 251–285. https://doi.org/10.1007/BF00120253