

PROPOSING A NOVEL APPROACH TO LEARN TO CODE THROUGH A GAMIFIED ONLINE SYSTEM BY CREATING MEANINGFUL INTERACTIONS

Research Paper

Mario Silic, Swiss School of Business and Management Geneva, Switzerland

Dario Silic, Swiss School of Business and Management Geneva, Switzerland

Dino Kolak, Swiss School of Business and Management Geneva, Switzerland

Marijana Leontic, College of Environmental Health and Safety, Croatia

Abstract

How can you engage and motivate students to learn to code in a more meaningful way to achieve better academic performance? We conducted an experimental study to propose a novel approach to learn to code through a gamified online system by creating meaningful interactions. Through the quantitative data collected in two countries with 159 students (experimental and control group) that was supported by interviews with 20 students, we found the student's performance is significantly improved in the experimental condition, but also that failure rates are much lower. We also identified a curvilinear relationship between mentorship support and the learning process. Our findings offer new insights on the importance of creating an environment that provides and drives engagement and motivation to support the learning experience. It is clear that motivation drives learning and learning is achieved only if it is really meaningful.

Keywords: gamification, programming, learn to code, teaching strategy, mentor

1 Introduction

A typical way to learn coding is the classroom setting, where teacher is the one driving the content, interactions and the entire learning process. Recently, we have witnessed the development of several new methods, ranging from massive open online courses (MOOCs) to boot camps which aim at teaching code by targeting different participant groups. All of these methods have their own challenges, ranging from MOOC's low completion rates and difficult interactions due to the high number of participants (Jordan, 2014) to mixed results offered by boot camps (Waguespack, Babb, & Yates, 2017). Since coding is becoming a new literacy, students increasingly want to learn to code. However, they also claim that they were not offered the opportunity, since it's the case for two-thirds of the 1850 surveyed Australian students (Microsoft, 2015). However, basic knowledge of programming is now part nearly of all engineering curricula (Topalli and Cagiltay, 2018). That is not

the case for all other disciplines such as, for example, business administration. Coding will clearly help students to become a better citizens tomorrow (Ball and Zorn, 2015).

One of the main challenges for novice programmers is that they usually do not know where to start, how to start and what to learn. Those facts are combined with the coding realities, which highlight the difficult of coding as a discipline, due to the many different concepts that must be acquired in a short amount of time (Rizvi, Humphries, Major, Jones, & Lauzun, 2011). It is also argued that the current way of teaching coding is simply outdated and should be reshaped (Makowsky and Zamansky, 2017).

In addition to these challenges, the motivation to learn seems to be one of the main roadblocks. This is because, without clearly motivated students, coding completion rates will remain low, as is the case for MOOCs. Students' grades will be impacted and there will be higher dropout rates and failures (Robins, Rountree, & Rountree, 2003). Different methods can be implemented to keep motivation high. Techniques, such as storytelling, seem to work well for children (Kelleher, Pausch, & Kiesler, 2007). Student motivation is usually different since students are not only motivated by grades. More importantly, there appears to be a disconnect between how students and teachers agree on what they should learn in the class, which drives their inner motivation (they see different kinds of learning situations and materials differently) (Milne and Rowe, 2002). Interestingly, in addition to motivation, another key challenge does not seem to be related to how to understand the basic programming concepts, but rather, how to apply them (Lahtinen, Ala-Mutka, & Järvinen, 2005).

Based on the literature review of these issues, we identify several challenges that we evaluate through the experimental study and review in the results and discussion sections: a) the student's motivation should be addressed by creating meaningful interactions; b) the teacher's role needs to be reshaped and changed to a mentor-based approach; c) a fully online learning to code platform should drive the immersive learning experience and d) novice students need gamification elements, such as feedback, progress, learning outcomes, etc. to drive their engagement.

In the next section, we summarize the literature review and the gaps that we identified.

2 Literature review

Past research suggests that providing easily accessible programming tasks (e.g., practical tasks in which "peer learning" is encouraged) to students, in such a form that some interactions and a graphical user interface are presented, can stimulate and motivate students to learn (Robins, et al., 2003). A game-like system could provide an opportunity for students to engage in new interactions which stimulate their curiosity, interest and motivation. This "student engagement" is a critical point, where student's buy-in is necessary. One way to accomplish this is through a problem-solving method. This

was found to be effective and meaningful, especially if students face problems by themselves and have to solve them on their own (Liu, Cheng, & Huang, 2011).

It is, therefore, important to determine how to leverage and drive the student's involvement and motivation. Immersion, flow and engagement were found to be important factors that could help students to learn (Hamari et al., 2016) through the use of gamification (a concept that originates from the Flow theory) elements (Barata, Gama, Jorge, & Gonçalves, 2013), especially in the online context with fully self-paced and collaborative online activities (McGrath and Bayerlein, 2013). It is clear that an engaging, immersive online learning gamified platform, with the objective of leveraging interactions and driving student's motivation, should positively impact the student's results and their overall learning experience.

An important component that can positively shape the learning experience, is instant feedback, which was already reported to be an important dimension in achieving flow (Rizvi, et al., 2011). Several other parallel improvements were studied by researchers, ranging from visual programming, narrative tools and functional programming approaches to object visualization in a 3D animation environment (Cooper, Dann, & Pausch, 2003). An interesting improvement that could positively impact the online experience, is the possibility of learning coding fully online. This means without having to install an offline compiler or any other libraries of frameworks. Impulsive learners can use the online compiler immediately, without the need to spend time doing it locally (Wolf, 2003). It could be done anytime and anywhere, since the compiler is always online.

Switching teacher's role to a more mentor-based (i.e., tutor or coach) approach was highlighted as the preferred approach in online learning (Beaudoin, 1990). However, resistance is still relatively high (Crosby, 2000). It has been found that despite the recent dramatic technological changes, we are still not fully maximizing the online opportunities. For example, in a traditional teaching setup, one teacher is teaching one programming language to one classroom. This can be seen as one closed environment. If this is changed to an online experience where the teacher's role is not to teach, but to drive, set goals, coach and assess goal-achievement, it would be an interesting path to explore. The objective would be to create meaningful engagement and motivation, by offering flexibility, availability, presence, collaboration, and a sense of community. This could boost the novice learning process (Lahtinen, et al., 2005). Advising students how to proceed to the next level by providing the direction and the instant feedback, while keeping the motivation and engagement high, could generate a new environment. In this atmosphere, in addition to grades, students would be motivated by the competition between themselves, while being fully immersed.

The research has provided mixed results. Some authors found that gamification does work in the educational context, while others did not find any support for better results after gamification was

introduced (Dicheva, Dichev, Agre, & Angelova, 2015). However, the initial results from this research stream are promising. They reveal that the right approach to gamification in education needs to address how to build the right technological support, conduct control studies and, above all, to fundamentally rethink the learning design (Kapp, 2012).

Based on this theoretical underpinning, we propose a new design for a learning to code course for novice users by introducing a mentor-based approach supported by a fully gamified platform. The objective of the platform is to create meaningful engagement and interactions to support the student's motivation. In the following sections, we present the research method as well as provide details about the design of the new coding course

3 Research Methodology

We conducted an experimental study during one semester in two countries (Switzerland and Croatia) among business administration students (novice coders) enrolled at the bachelor level in both universities. During their third year of studies, students are required to take an introductory course in computer programming language (in the python language). In both universities, students follow the standard setup, where they are taught the typical coding concepts, such as variables, problem-solving skills or debugging methods.

The programming course is mandatory and graded. It consists of small programming tasks that must be accomplished throughout the semester. Students are also required to deliver a group or an individual-based final programming project.

We conducted the experiment in two countries (Switzerland and Croatia) in order to increase the study validity and improve the overall generalizability.

3.1 Standard course format

A standard course is offered in the third year of the business administration program at both universities. It follows a traditional teaching method in which different programming concepts are introduced to students (including variables, loops, functions, arrays, etc.). The course is taught in one semester, in a 15-week format with three hours of teaching per week. Of these 45 teaching hours, 10 hours are for the theoretical portion and 35 hours are reserved for laboratory exercises, during which students must solve various programming tasks introduced in the theoretical portion of the course. The course helps students to acquire critical and analytical thinking skills by solving programming tasks and debugging programs.

The grading scheme for the course is composed of the lab exercises (30% of the final grade), a group/individual project (20%) and the final exam (50%). The final grade is the total sum of the points

that is translated into the grade as shown on Figure 1. (note: for Switzerland the grades are inversed – e.g., Fail(1) corresponds to 6).

<u>Final grade</u>	
91 – 100 points	Excellent (5)
81 – 90 points	Very Good (4)
71 – 80 points	Good (3)
61 – 70 points	Acceptable (2)
< 60 points	Fail (1)

Figure 1 Grading scheme

In order to pass the course by the end of the semester, students must have all components of the grade positively evaluated. The failure rate is an average of around 33%, with relatively high withdrawal rates of around 25% (i.e., withdrawal is not counted in the failure rate). The failure rate is mostly explained by the fact that students are not practicing enough on their own and easily lose their motivation. They also do not get support outside of classroom hours. As a result, it is very difficult for them to complete the group project. The non-completion of the group project and the difficulties in applying the principles that are learned, explains the high failure rates. However, this seems to be the negative result of a decrease in the student's motivation as the course is progressing and the difficulty is increasing.

3.2 Gamified course format

To address these challenges with the standard course format, we built a new online learning system in which we introduced several new concepts and enhancements:

- 1) Teaching as a Service (TaaS): we modified the role of the teacher to become a mentor (i.e., coach or content advisor)
- 2) Self-regulated learning: students learn on their own following an anytime, anywhere and anyplace concept. Classes were no longer provided. Everything was happening online, where students were teaching on their own.
- 3) Community building: to create meaningful motivation and interactions, we used social collaboration tools to focus on motivational aspects but also to bring a sense of community.
- 4) New coding platform: we created a new central place where the whole learning process was happening. The entire platform was fully gamified.
- 5) Grading remained the same and consistent between the two courses.

- 6) Project oriented goals: students were guided by their mentor throughout the project lifecycle.
- 7) Learning analytics: more descriptive feedback was introduced in which students were informed about their current performance and possible areas for improvement.

To leverage the motivational side, we introduced several gamification elements (Table 1).

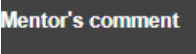
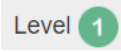
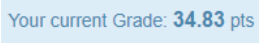
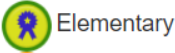
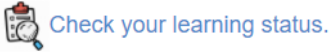

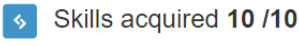
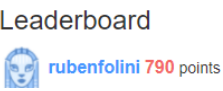

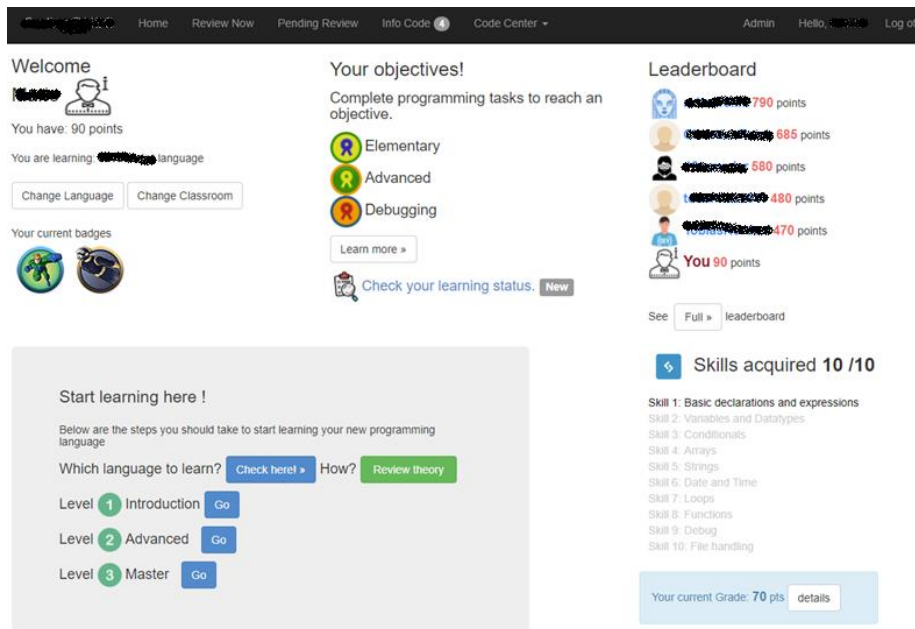
Gamification element	Proposed implementation
Instant feedback	
Levels	
Grades	
Objectives	
Learning status	
Badges	
Skills	
Leaderboard	
Avatar	

Table 1. Gamification elements introduced

3.3 Figure 2. shows how the new system user interface looks with the different gamification elements that can be found.



3.4

Figure 2. The new system

Similar to the standard classroom format, students also solved the predefined tasks that were adapted to the online context so that students were provided with tips (e.g., links) on how to solve different programming tasks.

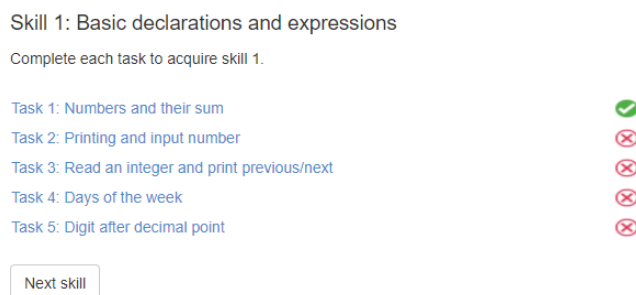


Figure 3. Example of a skill

The difficulty of the tasks was aligned with the student progress. The more the student progressed with his/her learning, the more difficult the tasks became. Each time the student submitted

a task through the online system, the mentor received an email stating that a new task was submitted for review. The task review was resulted in: a) an info code – meaning that task was solved correctly and a comment is sent back to the student; b) completed – mentor reviewed the task and accepted it and c) revise: the student had to revise the task. All these statuses (Figure 4) were displayed to the user on the main screen. Student could then revise the task or simply acknowledge the mentor’s feedback. Every submitted task automatically brought +10 points to the student that were either confirmed by the mentor during the review process or removed, if the task was not successfully completed. The mentor was also able to give fewer points, if the task code was not as expected but still worked.

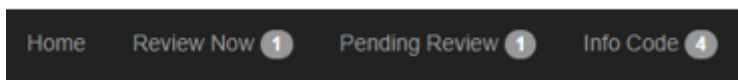


Figure. 4. Task statuses

This new approach was intended to create meaningful interactions motivate group learning, social engagement, and mentor-to-individual interactions. At the core of the new system, were interactions between the students and the mentor.

Figure 5 presents different interactions between the mentor and the students that include: a) instant code feedback; b) regular video conferences; c) live events (kick-off event held in the beginning of the semester to explain the mechanics and two other events in the middle of the semester); d) online collaboration through Slack.

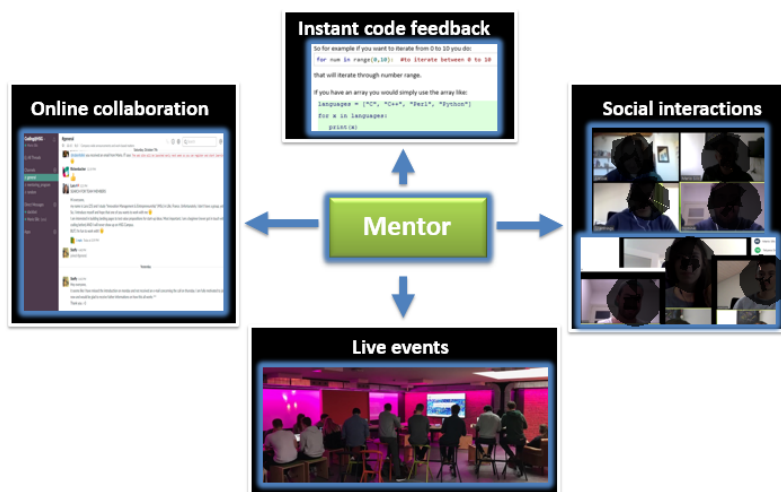


Figure. 5. Interactions

Similar to the standard course format, each student (or a group) had to deliver a final project.

3.5 Participants

In September 2017, we started the experiment with four student groups which included: one control group of students in Switzerland; one control group in Croatia that followed the standard approach (the one currently in place); one experimental group in Switzerland and another in Croatia, where both groups followed the new course format, as described in Section 3.2. In total, we had 74 students in the control group for both countries (35 control students in Switzerland and 39 in Croatia) and 85 for the experimental condition in both countries (48 for Switzerland and 37 for Croatia). Compared to the recent study by Topalli and Cagiltay (2018), which had 48 students in the experimental group, we believe that our sample is satisfactory to provide some statistically meaningful results.

To ensure the comparability between different groups, we employed several methods. We checked the students’ prior programming knowledge (we excluded seven students from the final results – 2 for Switzerland and 5 for Croatia, since they affirmed that they had already some prior programming knowledge), we compared demographics including gender differences, their CGPA (since in Switzerland, the grading scale is over 6 points we converted all CGPA to a 5-point scale).

Table 2 provides the participants demographics. Overall, 66% and 64% of students in the control group were male for Switzerland and Croatia respectively. Similar numbers are observed for the experimental group. We conclude that gender distribution is similar between the two groups, but also between the two countries.

As for the CGPA, we conducted an independent t-test where the result was not significant – $t(159) = 0.469$, $p = 0.549$. This means that students’ samples for both treatments (control and experimental) are similar, since their average CGPA scores are similar (experimental group: $M=1.65$; $SD=0.90$; control group: $M=1.70$; $SD=0.92$ with 0.05 confidence interval). We also did the same comparison between the countries to make sure that we could combine the samples and have similar groups. The T-test result was not significant $t(83) = 0.481$, $p = 0.629$ for Switzerland and $t(76) = 0.421$, $p = 0.532$ for Croatia.

Control Group					
Country	Gender				Total
	F		M		
Switzerland	12	34%	23	66%	35

	Croatia	14	36%	25	64%	39
						74
Experimental Group						
Country	Gender					
	F		M		Total	
Switzerland	15	31%	33	69%	48	
Croatia	13	35%	24	65%	37	
					85	

Table 2. Demographics for control and experimental groups

Based on this analysis, we posit that students in the experimental group will perform better than the students in the control group. This is because their motivation to learn will be much higher than in the standard format. We also argue that the failure rate for the overall course will be much lower, as result of the gamification elements and in particular, from the mentor’s feedback and the active engagement. Finally, we believe that interactions during the semester will have a curvilinear relationship for the learning curve, since we expect that mentorship will be less needed, since students will be progressing in their learning path. Thus, we hypothesize:

H1. Students in the experimental group will perform better than the students in the control group

H2. Failure rates in the experimental group will be lower than the failure rates for the control group

H3. The learning curve will be curvilinear, where mentorship need will decrease with the learning increase in the experimental group

To assess the above hypotheses, we used the same assessment method for both groups and both countries to minimize any possible bias. The same scoring system was used to assess the H1, failure rates were tracked for H2 and we used analytic possibilities of the social collaboration tools (e.g., Slack) to get the data on the interactions taking place within the experimental group. In parallel, we also surveyed 20 students (10 in each country) to get some qualitative insights about the experiment and, in particular, regarding the hypotheses..

4 Results

To compare students' results between the two groups (experimental and control) and answer the first hypothesis, we used an independent t-sample test. The test revealed a significant result for the student's final grade when comparing the two groups - $t(159)=2.95$, $p=0.002$ with ($M=1.82$, $SD=1.29$) for experimental group and ($M=1.22$, $SD=1.28$) for the control group, but also when comparing on a country level $t(83)=2.86$, $p=0.001$ for Switzerland and $t(76)=2.77$, $p=0.002$ for Croatia. The assessment for both groups was done in the same way, with the same content for the tasks that were graded and also for the final exam. Consequently, we conclude that Hypothesis 1 is supported.

The overall completion rate for the course was 92%, with only 8% failures for the experimental group. The control group failure rate was 25%. We calculated a Z score (2.243), which revealed that the result is significant at $p < 0.05$ (assuming two-tailed hypothesis test). Similar results were observed in both countries: Switzerland 90% of completion rate vs 93% for Croatia. When we surveyed 20 students (10 in each country), we specifically asked their opinion on why they failed. Several of the interviewees provided the following reasons: they did not have the motivation to continue until the end of the course (control group), the tasks were difficult to solve and required too much time (control group) or there was not enough time to start the course (experimental group). For example, one participant in the control group said: "for me the problem was that I was behind in the other courses and did not have motivation to work for 15 weeks on this course" and another student stated "I think the main issue for me was that group project was too difficult to complete – so I gave up". When we asked the participants who succeeded in the course, the main reasons for their success, they highlighted the following: 1) mentor's instant feedback followed by gamification elements (e.g., leaderboard) for the experimental group and 2) the structure of the course had a good balance between theory and practice (control group). Several participants in the experimental group highlighted the value of the new approach through the gamified platform. For example, one commented: "...not only it is great fun to solve your tasks, but the whole website and structure is thought through so well...and the result is a great project all around...". Another one added: "I would like to thank you [mentor] for initiating this great coding course and all the support you provided throughout the semester! It is really great to see such an engagement for students' learning progress.". When we asked interviewees if they were considering dropping the class or thought they would fail it, we received interesting insights. One commented that "yes, when I started I thought I would fail, since the final project sounded way too difficult for me, but my amazing mentor's feedback pushed me, and kept me motivated to go until the end" [experimental participant]. Another one added "the great thing is that I was able to start four weeks later – I'm not sure how I would manage this class and all tasks if I had to do it from the beginning – the anytime nature, enabled me to start later...that is what was great" [experimental

group]. Overall, we conclude that Hypothesis 2 is supported, not only through the decreased failure rates, but also through the student feedback which indicates that the gamified platform kept their motivation and engagement high.

Finally, in order to understand the learning path behavior, we gathered all of the interactions that occurred online. All the online interactions are associated with the mentor-to-student information exchange, where students asked the mentor to guide them through the course and programming tasks. Overall, the interactions' objective was to drive motivation through the mentor's feedback. In total, there were 1091 interactions recorded on the system (an interaction includes any communication exchange between the mentor and student). As it can be seen in Figure 6, the weekly interactions reached a peak, one month after the start of the course. This suggests that in the beginning, as the student learning was still in the initial phase, the need for the mentor was high. Since the students were learning through a self-regulated process, progressing through the tasks and advancing their knowledge and the skills, there was less of a need to interact with the mentor. This process reflects a curvilinear relationship between the learning process and the mentorship. Mentorship need clearly decreases with a learning increase. We also checked this with interviews, since we asked participants how they found their progress during the online course (experimental group) and the need to be supported. They clearly indicated that they needed the mentor's support intensively in the beginning. However, after some time, as they followed the self-regulated pace, that need was much lower. One participant highlighted "I think I contacted the mentor 20 times in the first two weeks, but then mentor showed me the right direction to follow and where to learn – so I got the habit on how to do it and did not need mentor anymore". Another one added "What I appreciated a lot is the great feedback on my submissions in the beginning – but as time passed, less I needed the support less. Somehow, I knew where to look for the solution." On the other hand, when the same questions were asked to the control group, the responses indicated that the teacher's support was needed more in a linear way. This meant that with time, the need for teacher's support was not declining, but the teacher's support was needed more. For example, as one participant explained this: "well, honestly, I didn't want to look around as it was simpler just to ask teacher to help me out". Another one added "I'm sure I could find the solution on my own, but probably I did like everyone else – I just asked for help".

Finally, we found that Hypothesis 3 is supported since mentorship need was clearly decreasing over time, as students were following the self-regulated pace and were more used to acting on their own.



5 Discussion and Conclusion

In this research, we have investigated how a gamified learning to code online system can improve student's performance through meaningful interactions and engagement across two countries. We found that student's performance is significantly improved in the experimental condition and failure rates are also much lower. Finally, we identified a curvilinear relationship between the mentorship support and the learning process. We conducted quantitative and qualitative studies to support our findings, which provide new insights into the importance of creating an environment that provides engagement and motivation to support the learning experience. Motivation clearly drives learning and learning is achieved, only if it is really meaningful.

In this context, we leverage the insights from past research that has indicated that games could engage students in learning programming concepts. (Fowler and Cusack, 2011). Another recent study showed that a small improvement of the course curriculum through real-life game development projects in the Scratch environment, can lead to higher student performance (Topalli and Cagiltay, 2018). We extend these by studies by suggesting that interactions combined with the mentor's role, instead of having a teacher's role, can significantly improve student's motivation and engagement which seem to be key elements in driving the student's performance or failure. The student's learning process and their problem-solving skills can clearly be leveraged through a different approach to learn, by changing the teacher's role to become more of a coach, leader, mentor or content advisor. At the same time, gamification elements, such as the leaderboard, seem to positively influence the student's learning curve so that their motivational level remains stable over time and does not decrease to lead to a negative result (e.g., failure or withdrawal).

Our study offers some interesting opportunities that should be explored further. For example, a better understanding of the psychological reasons that maintains a student's motivation consistent

over time, could provide some new and interesting insights about the student's psychological functioning. In addition, further understanding of other aspects of the learning process, such as different motivational states and what drives them, once a mentor is no longer needed to drive the engagement, could be an interesting path to explore. The study is also limited since we ran the experiment exclusively in the business administration department and we did not involve other departments, which could provide a different perspective. In addition, since the gamification platform offered something that was completely new and different, there is a possibility that student engagement was partly driven by the novelty effect of the platform. They might react differently, if the platform was not as novel.

In conclusion, our study offers an interesting new perspective on how to drive meaningful engagement and motivation through an online gamified platform, where students through a self-regulated concept, displayed better performance and lower rates of failure. In addition, by introducing the mentor driven learning process, the relationship between learning and the mentorship need became curvilinear, which positively supported the student's knowledge acquiring process

6 References

- Ball, T., & Zorn, B. (2015). Teach foundational language principles. *Communications of the ACM*, 58(5), pp. 30-31.
- Barata, G., Gama, S., Jorge, J., & Gonçalves, D. (2013). *Engaging engineering students with gamification*. Games and Virtual Worlds for Serious Applications (VS-GAMES), 2013 5th International Conference on.
- Beaudoin, M. (1990). The instructor's changing role in distance education. *American Journal of Distance Education*, 4(2), pp. 21-29.
- Cooper, S., Dann, W., & Pausch, R. (2003). *Teaching objects-first in introductory computer science*. ACM SIGCSE Bulletin.
- Crosby, R. H., Joy. (2000). AMEE Guide No 20: The good teacher is more than a lecturer-the twelve roles of the teacher. *Medical teacher*, 22(4), pp. 334-347.
- Dicheva, D., Dichev, C., Agre, G., & Angelova, G. (2015). Gamification in education: a systematic mapping study. *Journal of Educational Technology & Society*, 18(3), p 75.
- Fowler, A., & Cusack, B. (2011). *Kodu game lab: improving the motivation for learning programming concepts*. Proceedings of the 6th International Conference on Foundations of Digital Games.
- Hamari, J., Shernoff, D. J., Rowe, E., Coller, B., Asbell-Clarke, J., & Edwards, T. (2016). Challenging games help students learn: An empirical study on engagement, flow and immersion in game-based learning. *Computers in Human Behavior*, 54, pp. 170-179.

- Jordan, K. (2014). Initial trends in enrolment and completion of massive open online courses. *The International Review of Research in Open and Distributed Learning*, 15(1)
- Kapp, K. M. (2012). *The gamification of learning and instruction: game-based methods and strategies for training and education* San Francisco, US: John Wiley & Sons.
- Kelleher, C., Pausch, R., & Kiesler, S. (2007). *Storytelling alice motivates middle school girls to learn computer programming*. Proceedings of the SIGCHI conference on Human factors in computing systems.
- Lahtinen, E., Ala-Mutka, K., & Järvinen, H.-M. (2005). *A study of the difficulties of novice programmers*. *Acm Sigcse Bulletin*.
- Liu, C.-C., Cheng, Y.-B., & Huang, C.-W. (2011). The effect of simulation games on the learning of computational problem solving. *Computers & Education*, 57(3), pp. 1907-1918.
- Makowsky, J., & Zamansky, A. (2017). Keeping logic in the trivium of computer science: a teaching perspective. *Formal Methods in System Design*, 51(2), pp. 419-430.
- McGrath, N., & Bayerlein, L. (2013). *Engaging online students through the gamification of learning materials: The present and the future*. ASCILITE-Australian Society for Computers in Learning in Tertiary Education Annual Conference.
- Microsoft. (2015). Microsoft inspires Australian students to start computer coding now. Retrieved Date from <https://news.microsoft.com/en-au/2015/05/15/microsoft-inspires-australian-students-to-start-computer-coding-now/>.
- Milne, I., & Rowe, G. (2002). Difficulties in learning and teaching programming—views of students and tutors. *Education and Information technologies*, 7(1), pp. 55-66.
- Rizvi, M., Humphries, T., Major, D., Jones, M., & Lauzun, H. (2011). A CS0 course using Scratch. *Journal of Computing Sciences in Colleges*, 26(3), pp. 19-27.
- Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer science education*, 13(2), pp. 137-172.
- Topalli, D., & Cagiltay, N. E. (2018). Improving programming skills in engineering education through problem-based game projects with Scratch. *Computers & Education*, 120, pp. 64-74.
- Waguespack, L. J., Babb, J. S., & Yates, D. J. (2017). *Triangulating Coding Bootcamps in IS Education: Bootleg Education or Disruptive Innovation?* Proceedings of the EDSIG Conference ISSN.
- Wolf, C. (2003). *iWeaver: towards' learning style'-based e-learning in computer science education*. Proceedings of the fifth Australasian conference on Computing education-Volume 20.

- Silic, M., & Back, A. (2014). Shadow IT—A view from behind the curtain. *Computers & Security, 45*, 274-283.
- Silic, M., Barlow, J. B., & Back, A. (2017). A new perspective on neutralization and deterrence: Predicting shadow IT usage. *Information & management, 54(8)*, 1023-1037.
- Silic, M. (2019). Critical impact of organizational and individual inertia in explaining non-compliant security behavior in the Shadow IT context. *Computers & Security, 80*, 108-119.
- Silic, M., Silic, D., & Oblakovic, G. (2016). Influence of Shadow IT on innovation in organizations. *Complex Systems Informatics and Modeling Quarterly CSIMQ, (8)*, 68-80.