Journal of Education Technology
Volume 6, Number 4, 2022 pp. 598-606
P-ISSN: 2549-4856 E-ISSN : 2549-8290
Open Access: https://ejournal.undiksha.ac.id/index.php/JET

IT Curriculum for Boot Camp: An Iterative Development In Applying OBE In Computer Science Education for Non-Formal Institution

Tony Wibowo[1*], Julia Veronica[2]
[1,2] Department of Computer Science, International Batam University, Batam, Indonesia

ARTICLE INFO

Article history:
Received August 12, 2022
Revised August 14, 2022
Accepted October 12, 2022
Available online November 25, 2022

Kata Kunci:
Kata Kunci: Desain kurikulum, boot camp, pemrograman

Keywords:
Curriculum design, boot camp, programming

DOI:
https://doi.org/10.23887/jet.v6i4.51343

ABSTRAK

Seiring berkembangnya teknologi dalam merancang software, pemrograman komputer hadir hampir di semua aspek bisnis modern serta dalam kehidupan sehari-hari. Pendidikan pemrograman kembali menarik perhatian dan keahlian yang diincar bagi tenaga kerja. Namun, para peneliti sebelumnya menemukan kesulitan yang dihadapi saat belajar bahasa pemrograman. Penelitian ini bertujuan untuk menganalisis perancangan kurikulum berdasarkan kepuasan konsumen dengan studi kasus pada startup Teknokasi. Penelitian ini menggunakan metode kualitatif dengan mengumpulkan data melalui survei. Metode yang digunakan untuk mengumpulkan data yaitu wawancara dan kuesioner. Dalam penelitian ini tercatat tiga iterasi dan pengembangan kurikulum masih perlu ditingkatkan pada penelitian selanjutnya. Iterasi tersebut menggunakan framework scrum dan OBE sebagai landasan teori. Dari hasil penelitian ini membuktikan bahwa masukan dari mentor dan siswa berpengaruh pada pengembangan kurikulum berikutnya dan dalam waktu singkat dapat mengasah kemampuan pemrograman khususnya dalam pemograman JavaScript. Efektifitas dari kurikulum ini perlu diteliti dalam peneltiian berikutnya.

ABSTRACT

With the development of technology in software design, computer programming is now prevalent in almost every aspect of modern business and daily life. A renewed focus on programming education is attracting workers' attention and converting skills. In previous research, researchers found difficulties encountered while learning programming languages. In this study, we conduct a case study on Teknokasi startup to develop a curriculum based on customer satisfaction. In this study, data was collected through surveys, which is a qualitative method. The methods used to collect data are interviews and questionnaires. The results of this study indicated three iterations, and further research is required to improve curriculum development. In this iteration, we will use the scrum framework and OBE as a theoretical base. Using input from mentors and students, this study was able to develop the next curriculum and improve programming skills, especially targeted Javascript expertise in a short time frame. Further studies need to be performed to assess this curriculum's effectiveness.

1. INTRODUCTION

Due to the increasing demand for programming skills and information technology, education in programming is again attracting considerable attention (Saito & Watanobe, 2020; Tellhed et al., 2020; Zhan et al., 2022). There is an enormous amount of work that goes into developing these programs, which requires a large number of programming professionals who understand how computer applications help people in modern times. Programming skills are not only the most in-demand skills today; in addition, programming skills are believed to enhance problem-solving, logical reasoning, and creativity (Scherer et al., 2020; Younis et al., 2021). Previous studies found that students face challenges when learning a programming language (Gorson & O'Rourke, 2020; Ivanov & Karsakov, 2021). It is a big challenge for students to bridge the gap between how they think and how computers think. Most programming tutorials emphasize how to practice certain functions and commands. They don't provide contextual information describing why and when these functions and commands should be used (Yang et al., 2018). To overcome this difficulty, choosing the best learning methods and tools is very important for people who are just starting to learn programming (Farshidi et al., 2021; Shen et al., 2019). However, there is no doubt that learning difficulties occur across all fields. The field of programming carries its own challenges due to the constant development of programming (Cagnoni et al., 2020; Yang et al., 2018). As with learning a language, programming requires a lot of practice (Alrashedy et al., 2020; Shen et al., 2019). It is also important to know how to resolve problems when they occur due to unknown errors in

** Corresponding Author: Tony Wibowo: tony.wibowo@uib.ac.id* programming. Due to the lack of mentorship, new students will find it difficult to learn programming (Lakanen & Kärkkäinen, 2019). Call education a place where one can shape their characteristics gradually (Ayuningtyas & Setyaputri, 2020). Education consists of three different paths: formal (school), non-formal (community/outside school), and informal (family). There are several levels of formal education in the formal education system, which is divided into kindergarten, elementary, junior high schools, high schools, and universities. As part of non-formal education, learning activities are offered by the community that support formal education in a variety of ways. In formal education, the basis of the curriculum is well-defined cognitive knowledge and all student receive the same formal education content. The curriculum for non-formal education is specifically designed to meet the needs and interests students, using tasks, and skills (Ahmad et al., 2022; Haerullah & Elihami, 2020).

Over the years, there have been numerous attempts to solve learning problems in the field of information technology. As of today, school curriculums include programming skills events for elementary students rather than just those with an interest in technology and information (Saito & Watanobe, 2020). Although the language has been adapted to meet the needs of novice programmers, novice programmers have trouble learning it (Cico et al., 2020; Falgenti, 2020). As a result, programming boot camps provide technology-focused training programs devoted to teaching programming, frameworks, systems, and tools to entry-level software developers (Waguespack et al., 2018; Younis et al., 2021). Many practitioners have held boot camps, but no studies have examinded how curriculum is developed for bootcamps using scrum framework methodology and user input for curriculum improvement. So, we use Teknokasi startup as a case study. The startup Teknokasi has a focus on education and technology. Specifically, Teknokasi provides the opportunity for novices alike to learn IT and become professionals. During a Teknokasi boot camp, students learn more than just coding. As a result, students are taught how to make products in everyday life. To create a product, students develop and share ideas to solve problems and collaborate. As support for this research, we present results from a previous study.

The design of this curriculum is based on previous research. To investigate programming learning difficulties, researchers used student attitudes and opinions to identify what they thought about programming in schools after and before the teaching and learning activities (Yang et al., 2018). Curriculums are designed by us according to what students will learn during class. In this study, it was found that most students do not understand programming, some have a misunderstanding of it, and when programming directly is taught to them, it can bring positive changes within a certain period. A recent study examined curriculum development for programming to meet the needs of students who are not computer science majors (Dawson et al., 2018). Programming learning outcomes improve when curriculum design is applied iteratively, especially for students who are not majoring in computer science. This study uses the OBE model as a basis for developing a real estate appraisal curriculum for determining curriculum success (Wu, 2021). Other research states that use qualitative methods for data collection to assess whether it is necessary to improve curriculums (Busetto et al., 2020). Other research discussed how feedback from stakeholders can be used to design and improve the course curriculum (Misra & Priyadarshini, 2018). In the study, a survey design was suggested to collect data on the course users and the module's benefits, as well as parameters necessary to measure program outcomes.

According to the background research in the above paragraph, we recommend a curriculum with an iterative and effective assessment of learning. This study aims to develop a curriculum design based on customer satisfaction with case studies on Technocation startups. Curriculum development aims to identify how to design an IT curriculum based on boot camp customer satisfaction, with a case study on Teknokasi startup using iterative development. It has the potential to fill the gap in programming learning for Teknokasi students if done correctly. Our state of the art is to recommend a curriculum with effective assessment tools for monitoring and improving student learning. The curriculum will repeated three times using scrum framework and outcome based education as a theory based. We will revised curriculum based on mentors and students from Teknokasi. Once all three rotations are compeleted, we will conclude our study.

2. METHOD

Research method consist of implementing concepts on how to implement the research to be carried out (Darmalaksana, 2020). In this study, we used qualitative methods in an iterative process, such as (Busetto et al., 2020). Using scrum as an iterative development model, this research also uses an applied method. In this study, we conducted a qualitative survey to collect data. Below, we discusses in more detail how the research was conducted. Curriculum design begins with formulating problems based on the topic raised to determine the direction and objective. Following that, we used the qualitative method, scrum framework, and OBE to develop the curriculum in an iterative approach before teaching and learning activities began. As part of the data collection stage, we identify the parameters to consider in designing a curriculum to address the programming learning gap for Teknokasi students. We develop a list of qualitative survey with open-ended questions for enhances teaching and learning by generating meaningful information (Braun et al., 2020; Story & Tait, 2019). These were presented to students as interviews and questionnaires, with the results used to develop the next curriculum, if needed. Considering the small size of the population, we use a census sampling method. After several cycles of curriculum improvement, we will conclude and provide recommendations to readers at the end of the third cycle. An outline of our curriculum design research flow is provided presented in Figure 1.

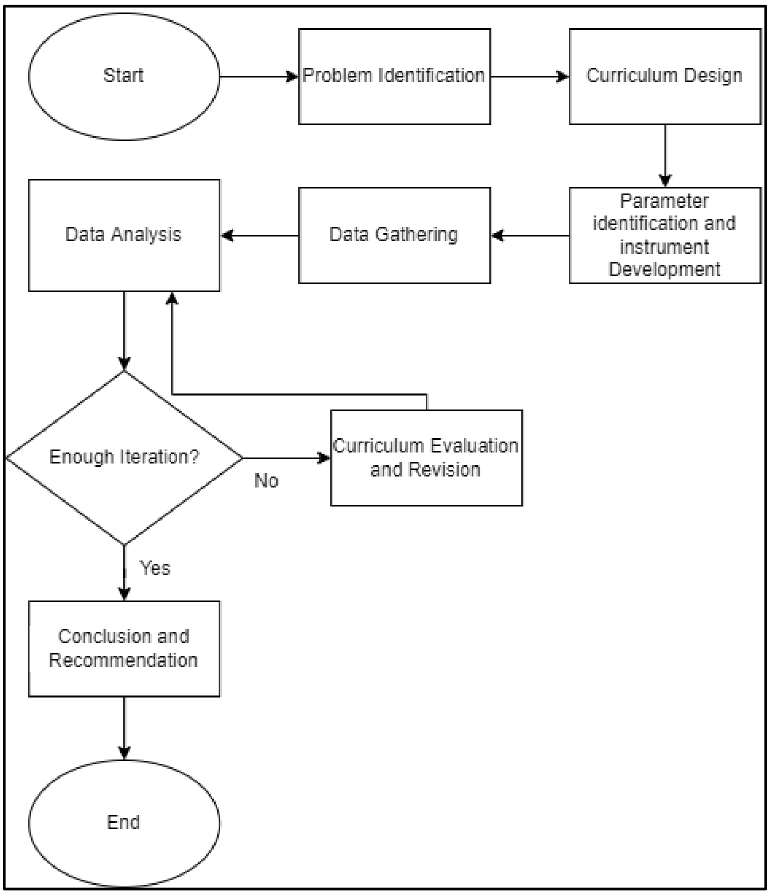
Start
Problem Identification
Curriculum Design
Parameter identification and instrument Development
Data Gathering
Data Analysis
Enough Iteration?
No
Curriculum Evaluation and Revision
Yes
Conclusion and Recommendation
End

Figure 1. Research Flow

After knowing problem identification, we design a curriculum. We chose JavaScript as a programming language to design the software engineering curriculum. To determine whether the curriculum was successful, we applied OBE (Outcome Based Education) as a theoretical base. In order to identify OBE in the Teknokasi curriculum as student learning, we determine learning outcomes, learning processes, and learning assessments (Prihantoro, 2020). Learning outcomes can be used to determine students' abilities that will be used during the learning process and for them to be able to present their products to obtain a coding certificate. Throughout the learning process, students gain an understanding of logic and problem-solving in programming. They also apply knowledge of mathematics, algorithms, computer theory, and computer systems principles. In theory and practice, students can understand the workflow of website creation by understanding both the client side and the server side, also the ability to communicate and project management.

A structured learning and assessment system curriculum will ensure that the learning process occurs by identifying the learning outcomes. In this curriculum, we focus on Sofware Engineering (SE), especially web-based application engineering, where students determine their future careers after attending the course, not what the experts recommend. The curriculum has been divided into five categories: design, coding, database, product management, and communication skills. From this category will come learning materials organized by level. To assess students' understanding, mentors provide feedback and assessments in report cards so students can improve in the future. The curriculum design process takes two weeks. The curriculum and course content is always up-to-date in curriculum design (Baham, 2019; Milićević et al., 2019; Salza et al., 2018; Subih et al., 2019). As part of iterative development, we use the scrum framework to develop the curriculum. As a result, the duration will vary based on the curriculum. However, it will not exceed a calendar month. The scrum framework sprint cycle diagram is presented in Figure 1. Scrum Execution showed in Table 1.

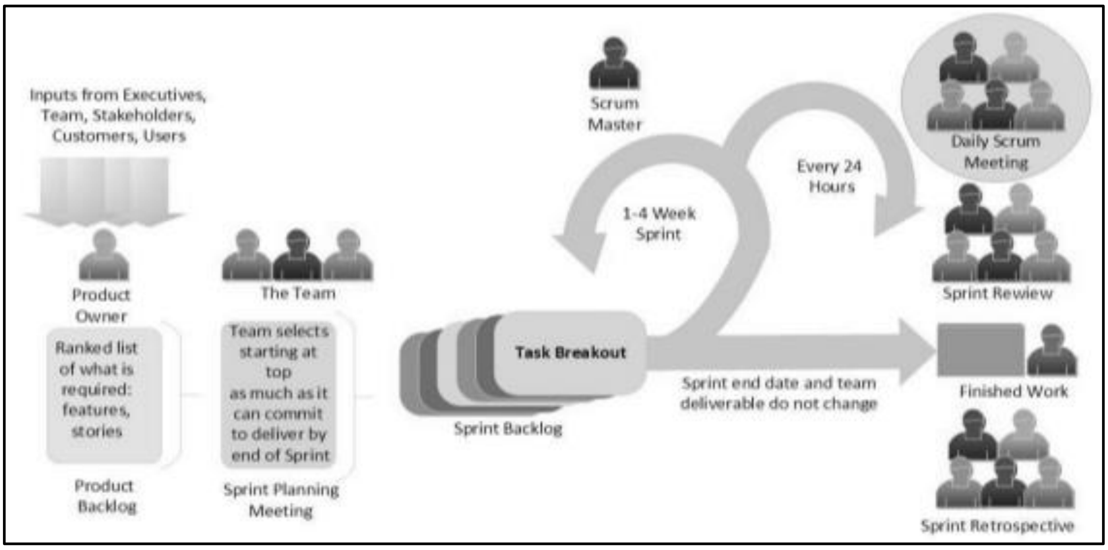
Inputs from Executives, Team, Stakeholders, Customers, Users
Product Owner
Ranked list of what is required: features, stories
Product Backlog
The Team
Team selects starting at top as much as it can commit to deliver by end of Sprint
Sprint Planning Meeting
Task Breakout
Sprint Backlog
Scrum Master
1-4 Week Sprint
Every 24 Hours
Daily Scrum Meeting
Sprint Review
Sprint end date and team deliverable do not change
Finished Work
Sprint Retrospective

Figure 2. Scrum Sprint Cycle

Table 1. Scrum Execution

Step	Description
Product backlog	We determines the product backlog in the first stage of the scrum method. We select material to teach based on level. The material is arranged in a product backlog that will later be solved according to level.
Sprint planning	As a second step, sprint planning involves deciding what plans will be implemented within the specified time frame.
Sprint backlog	This stage involves distributing a list of sprint backlog items that are planned for completion within a certain time period. When the project process is carried out, the sprint backlog makes things easier because everything is arranged neatly and carefully.
Daily scrum	When the product backlog, sprint planning, and sprint backlog have been determined, we move on to distributing the backlog work, which distributes material work according to predetermined levels. As part of the scrum team, the product owner and Scrum master will provide regular guidance as to the progress of the product backlog.
Sprint review	Sprint reviews occur in the fifth stage and involve re-checking with the scrum team, namely the product owner and scrum master, to see if anything has been missed or neglected from curriculum design. In addition, a sprint is conducted to evaluate the curriculum design and optimize the learning objective if necessary. After that, the level of learning was readjusted. We implement the curriculum into teaching and learning activities, so Teknokasi students will receive materials according to their level when the curriculum is complete.
Sprint retrospective	To determine whether there are any gaps in the curriculum, we conducted a sprint retrospective with the help of a product owner and scrum master. A gap will be found when students fill out surveys and conduct one-on-one interviews as part of the teaching and learning process. The curriculum will then be repeated in the following rotation if needed. Following the student feedback, we develop a curriculum and then retest the students.

To improve the teaching and learning process, we collect data using a qualitative method after completing the curriculum design (Steyn et al., 2018). A qualitative survey was conducted among students in Teknokasi by interviewing and asking them to complete questionnaires and obtaining their opinions rather than the researchers. Due to the small population, we used census sampling by selecting all Teknokasi students from the class as a sample (Alafgani & Purwandari, 2019). A second step is to identify the parameters necessary to overcome the difficulties of learning IT, namely satisfaction and exploration of students' learning attitudes before and after participation. Whether or not curriculum improvements are necessary will depend on the feedback.

3. RESULT AND DISCUSSION

Result

Developing a curriculum at Teknokasi, using the scrum framework to iteratively improve the curriculum based on Outcome Based Education (OBE). Through OBE, we can determine the results, processes, and assessments of student learning. Javascript programming language is the main topic of curriculum design in the area of software engineering. We design boot camps based on programming skills that all students must achieve to complete the course. During this course, they will be prepared to become a full-stack javascript web developer. A Teknokasi boot camp has two learning objectives: (1) increasing students' interest in programming and (2) improving students' understanding of programming. There are five learning areas that students will study, specifically design, coder, database, product management, and communication. Each of the five categories is interconnected with the others. Using these five categories as a guide. Students will learn about application design, front and backend workflow, logic, problem-solving skills, and communications. As a result, we determine the learning materials that help students focus and achieve the goals of Teknokasi boot camp. There are different learning materials at each level, as can be seen from the image above. In level one, students learn more about web development, web layout and Figma, basic HTML/CSS, javascript programming, git, and basic communication skills. It takes 90 minutes to complete level one of the training with three classes a week. During the week following the class implementation, there will be ten meetings and one day's expo. Students at level two will master intermediate javascript and become familiar with the framework. There will be a strong focus on user interfaces and user experience, intermediate HTML/CSS, intermediate javascript, javascript framework, MySQL database using sequelize, product management (scrum), product deployment (hosting), and presentation skills. In the second level of learning, the class meets five times a week for 1 hour and 45 minutes. There will be fourteen class meetings plus an expo the following week. In level three, students master advanced javascript to deepen their understanding of the framework. They also learn advanced UI/UX, advanced HTML/CSS, advanced javascript, intermediate javascript framework, intermediate database, test-driven development with jest, and business presentation skills. It takes 1 hour and 45 minutes to complete the third level of learning with five classes a week. There will be fourteen meetings plus a one-day expo the following week. The fourth level explains how projects and programming work together. In level four, students have not studied design but rather developed their programming skills. It will teach students javascript experts, advance, framework, database, and project negotiation. It takes one 1 hour and 45 minutes to complete the fourth level of learning with five classes a week. There will be fourteen class meetings plus a one-day expo the following week. After the curriculum was created, we explained the curriculum creation to the mentor and scrum master based on the OBE identification. After mentors and scrum masters provide feedback on the curriculum, if deficiencies are apparent before learning activities, the product backlog will be revised under the scrum method process.

The a mentor report regarding curriculum evaluation are: (1)Add two more levels where students first learn basic logic, then move on to problem-solving; (2) Add grooming and CV units at all levels for the communication skills category; (3) Learning logic and problem-solving for the basic level takes about one month. After incorporating mentor input into the first rotation, we enhanced the curriculum in the second rotation using scrum and OBE. Regardless of the level of participation, the learning objectives of the five-unit categories are the same. In the design category, we add Canva at level one and Figma at level two, while in the coder unit category, code.org is added at level one and scratch at level two. In communication skills, grooming and CV writing are included. Before learning programming languages, a student in level one studies programming logic. The first level of learning includes Canva, code.org, grooming, and CV building. Students attend two sessions every week for one week. It took eight meetings in total for the class. Level one consists of a written exam because students have not touched programming and are still practicing computer introductions and logic (theory-based) to apply programming. On level two, students learn the basics of Figma, scratch, grooming, and improving their CVs. The second level of learning lasts sixty minutes with classes held twice a week, plus a one-day expo the following week. There is no difference in the material and duration of learning levels three to six, but there are additional materials regarding communication skills, such as the second rotation image. In the next step, we show the mentor and scrum master to get feedback. After that, they ask us todo a test with the students. To implement the curriculum that has been designed, we open the class on the day of the batch level registration opening. A schedule and permission to interview students were created from this implementation after the batch-level participants were met. A week before the expo, we provided information about interviews via social media to students, followed by a third day before and during the expo. Interviews will be conducted after the expo. However, the questionnaire was filled out 15 minutes before the end of the expo. As a result of this implementation, we received ten feedbacks from students and mentors. According to the qualitative survey results, three things need to be added to curriculum design: a learning timeline, assigning tasks to students and sharing theory and practice. The CV units of communication skills have changed in curriculum design. Levels one and two emphasize personality grooming, while level three focuses on CV creation. Following is the result of using scrum and OBE to design the third rotation curriculum. Curriculum Third Rotation showed in Figure 3.

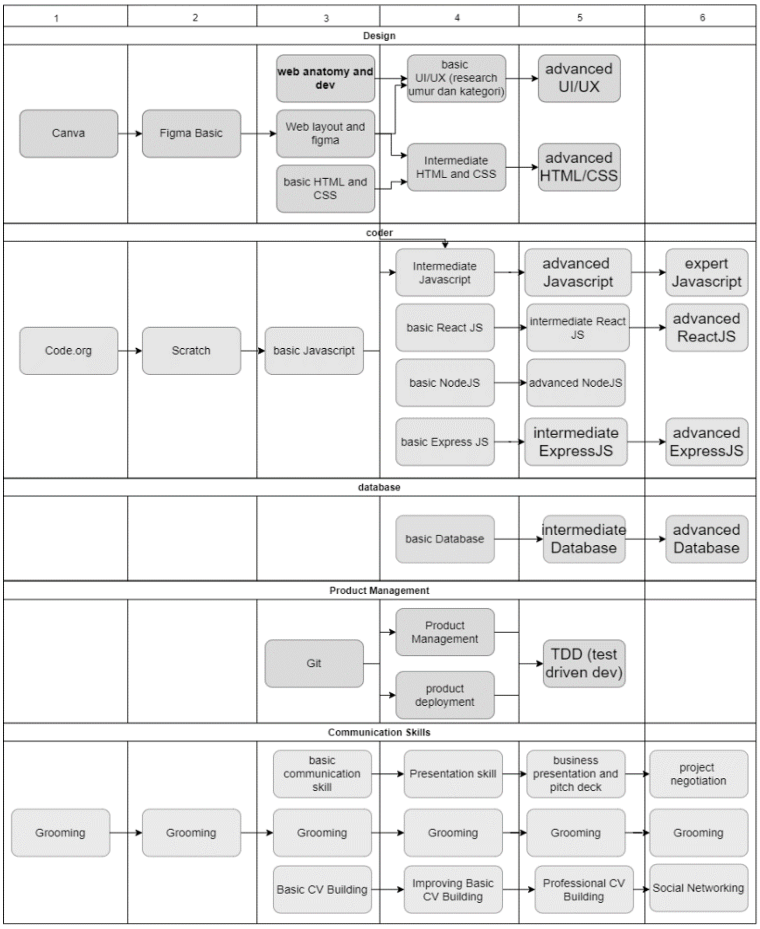

Figure 3. Curriculum Third Rotation

It can be seen that the category of communication skills has been changed, but the material and time spent learning remained the same in the second rotation. We calculate the average amount of theory and practice students receive for each unit per level. The ideal amount of theory is 30%, and the optimal amount of practice is 70%. The following table shows the levels of theory and practice. Average Theory and Practice Each Levels showed in Table 2.

Table 2. Average Theory and Practice Each Levels

Level	Theory	Practice
One	40%	59%
Two	44%	56%
Three	35%	64%
Four	31%	69%
Five	27%	73%
Six	30%	70%

According to the timeline, we determine the strategy for giving assignments to students. In addition to the type of task, we determine whether students work individually / in groups / in class / at home, as well as the learning unit assigned the task and the assignment strategy. To complete curriculum third rotation creation, we explain OBE identifications to the mentor and scrum master. The goal is to receive feedback from mentors and scrum masters. Unlike the first and second rotations, the third rotation does not provide ask for mentor feedback nor does it open the class to student feedback. Instead, it favors improvement and enhancement.

Discussion

In, MOOCs, the focus is on improving teaching rather than providing open access to higher education (Calvo et al., 2019; Zhang et al., 2019). Meanwhile, boot camp is non-formal event that everyone can attend, regardless of level education. In recent years, both of this learning methods have received considerable attention. Mentors are necessary when learning engineering skills such as coding. Students in MOOCs can access mentors, but they can't contact them directly, unlike students in boot camps, who can directly meet face-to-face with mentors. As a result, many entry-level software developers seek out programming boot camps that teach programming, frameworks, systems, and tools. They teach people without a programming background how to build and deploy applications (Ihantola et al., 2013; Waguespack et al., 2018). In addition to duration, depth of learning, curriculum, and programs, programming boot camps have varying relationships with employment opportunities (Kwon et al., 2020; Price & Dunagan, 2019). In the end, everything comes back to the courses that the students will follow. It is determined that the learning materials, level of difficulty, and student's abilities will determine how long the students will spend learning.

As we can see, from first to third rotation curriculum design shows that the design influences student learning outcomes. The development of curriculum through iterative development reduces the learning gap for Teknokasi students in programming. Through an up-to-date curriculum, we can ensure students do not experience the same problems (Busetto et al., 2020). With the available levels, students can practice their programming skills in short lessons. Aside from programming, they also learn design, database, product management, and communication skills. As a result, the five categories become a single unit called programming expertise. The content is divided into relatively small sections with short-term and easy-to-understand learning by using bite-sized learning in a boot camp (KOH et al., 2018; Stevanović et al., 2019). An IT curriculum should be made by a specialist. As a result, people do not have to learn many different things but rather more specific skills such as programming. Due to the development of specialization and the need for unique skill sets, this development is also driving an increase in skill set demands. Learning boot camp is complemented by project-based learning. The use of projects makes learners adapt more quickly to the learning process. It can be challenging for learners to keep up with the industry's demands with case studies and controlled projects. While it makes learning harder for some learners, for the majority, it is not impossible (Krajčovičová & Cápay, 2012; Shen et al., 2019). The OBE provides a theoretical basis for determining student goals, learning outcomes, and assessments to improve results through learning analysis (Rawlley & Mehra, 2020). In a short time, students will be able to learn about full-stack JavaScript web development targets and strategies (Wu, 2021). Also based on mentor and students' input, have directed to develop a skillset within a short period of time. As a result, we hope that this research can help educational institutions implement programming curriculums (Dawson et al., 2018; Misra & Priyadarshini, 2018; Yang et al., 2018). Using the JavaScript programming language as an example, it aims to reduce students' learning difficulties in programming. The ability to code will also be a necessary skill in the future. As a drawback, this research only focuses on preparing learning materials that students will accept. In the next study, we will examine the curriculum's effectiveness. Due to deficiencies in the curriculum, we didn't conduct a qualitative survey during the first rotation to get feedback from students. The mentor requests two new levels and units to be taught to students in which they master somethings never before taught in school. In the second rotation, after developing a curriculum based on feedback, the class can be opened and receive ten feedback from mentor and students. The ten qualitative survey results above suggest that three components need to be added to the curriculum design: making a learning timeline, assigning tasks, and sharing theory and practice. According to the third rotation study, the next rotation study addressed some drawbacks about this curriculum's effectiveness based on customer satisfaction.

4. CONCLUSION

Our study of curriculum design and development in Teknokasi startup draws some conclusions. According to this study, programming is a skill that is necessary and should be further developed to overcome the learning gap. Programming education must go beyond formal education and should be honed through non-formal education. Mentor and student feedback are also important for designing a better curriculum. By reducing students' learning difficulties, especially with JavaScript programming language, curriculum improvements are crucial when gaps are found before teaching and learning activities begin.

5. REFERENCES

Ahmad, A., Madani, F., Ishaq, M., Purwito, L., & Sari, R. P. (2022). Evaluasi Kebijakan Merdeka Belajar Pada Satuan Pendidikan Nonformal. *Aksara: Jurnal Ilmu Pendidikan Nonformal*, *8*(2).

https://doi.org/10.37905/aksara.8.2.1143-1154.2022.

Alafgani, M., & Purwandari, E. (2019). Self-Efficacy, Academic Motivation, Self-Regulated Learning and Academic Achievement. *Jurnal Psikologi Pendidikan dan Konseling: Jurnal Kajian Psikologi Pendidikan dan Bimbingan Konseling*, *5*(2), 104. https://doi.org/10.26858/jppk.v5i2.10930.

Alrashedy, K., Dharmaretnam, D., German, D. M., Srinivasan, V., & Gulliver, T. A. (2020). SCC++: Predicting the programming language of questions and snippets of Stack Overflow. *Journal of Systems and Software*, *162*. https://doi.org/10.1016/j.jss.2019.110505.

Ayuningtyas, P., & Setyaputri, N. Y. (2020). Ajian Jaran Goyang Mobile Apps: Sarana Pendidikan Karakter Siswa secara Virtual Melalui Sinergitas Pendidikan Formal dan Non Formal. *Prosiding Seminar Nasional Penalaran dan Penelitian Nusantara*, *1*, 117–129.

Baham, C. (2019). Teaching Tip: Implementing Scrum Wholesale in The Classroom. *Journal of Information Systems Education*, *30*(3), 141–159.

Braun, V., Clarke, V., Boulton, E., Davey, L., & McEvoy, C. (2020). The Online Survey As A Qualitative Research Tool. *International Journal of Social Research Methodology*, *24*(6), 641–654. https://doi.org/10.1080/13645579.2020.1805550.

Busetto, L., Wick, W., & Gumbinger, C. (2020). How to Use and Assess Qualitative Research Methods. *Neurological Research and Practice*, *2*(14). https://doi.org/10.1186/s42466-020-00059-z.

Cagnoni, S., Cozzini, L., Lombardo, G., Mordonini, M., & Poggi, A. (2020). Emotion-based analysis of programming languages on Stack Overflow. *ICT Express*, *6*(3). https://doi.org/10.1016/j.icte.2020.07.002.

Calvo, S., Morales, A., & Wade, J. (2019). The use of MOOCs in social enterprise education: an evaluation of a North–South collaborative FutureLearn program. *Journal of Small Business and Entrepreneurship*, *31*(3), 201–223. https://doi.org/10.1080/08276331.2018.1453241.

Cico, O., Jaccheri, L., & Duc, A. N. (2020). Towards Designing an Experience-based Course around Innovation Bootcamps - A Cohort Study. *Proceedings - Frontiers in Education Conference, FIE*, *2020-Octob*. https://doi.org/10.1109/FIE44824.2020.9274225.

Darmalaksana, W. (2020). Metode Penelitian Kualitatif Studi Pustaka dan Studi Lapangan. In *Pre-print Digital Library UIN Sunan Gunung Djati Bandung*.

Dawson, J. Q., Campbell, A., Allen, M., & Valair, A. (2018). Designing An Introductory Programming Course to Improve Non-Majors' Experiences. *SIGCSE 2018 - Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, 26–31. https://doi.org/10.1145/3159450.3159548.

Falgenti, K. (2020). Studi Komparatif Program Visual Dinamis untuk Pembelajaran Algoritma dan Pemograman Berorientasi Objek. *Journal of Applied Computer Science and Technology*, *1*(1). https://doi.org/10.52158/jacost.v1i1.53.

Farshidi, S., Jansen, S., & Deldar, M. (2021). A decision model for programming language ecosystem selection: Seven industry case studies. *Information and Software Technology*, *139*. https://doi.org/10.1016/j.infsof.2021.106640.

Gorson, J., & O'Rourke, E. (2020). Why do CS1 Students Think They're Bad at Programming?: Investigating Self-efficacy and Self-assessments at Three Universities. *ICER 2020 - Proceedings of the 2020 ACM Conference on International Computing Education Research*, 170–181. https://doi.org/10.1145/3372782.3406273.

Haerullah, & Elihami. (2020). Dimensi Perkembangan Pendidikan Formal dan Non Formal. *Jurnal Edukasi NonFormal*, *1*(1), 199–207.

Ihantola, P., Helminen, J., & Karavirta, V. (2013). How to study programming on mobile touch devices: interactive python code exercises. *In Proceedings of the 13th Koli Calling International Conference on Computing Education Research*. https://doi.org/10.1145/2526968.2526974.

Ivanov, E., & Karsakov, A. (2021). Visual programming language for data visualization based on visual grammar. *Procedia Computer Science*, *193*. https://doi.org/10.1016/j.procs.2021.10.041.

KOH, N. S., GOTTIPATI, S., & SHANKARARAMAN, V. (2018). Effectiveness of Bite-Sized Lecture on Student Learning Outcomes. *In 4th International Conference on Higher Education Advances (HEAD'18)*, 515–523. https://doi.org/10.4995/head18.2018.8027.

Krajčovičová, B., & Cápay, M. (2012). Project Based Education of Computer Science Using Cross-Curricular Relations. *Procedia - Social and Behavioral Sciences*, *47*, 854–861. https://doi.org/10.1016/j.sbspro.2012.06.747.

Kwon, C., Cao, H., & An, Y. (2020). Research on the Training Model of High-Tech Talents Based on the Support of Public Sector. *Advances in Economics, Business and Management Research*, *118*(Msie 2019), 275–280. https://doi.org/10.2991/msie-19.2020.57.

Lakanen, A. J., & Kärkkäinen, T. (2019). Identifying Pathways to Computer Science: The Long-Term Impact of

Short-Term Game Programming Outreach Interventions. *ACM Transactions on Computing Education*, *19*(3), 1–30. https://doi.org/10.1145/3283070.

Milićević, J. M., Filipović, F., Jezdović, I., Naumović, T., & Radenković, M. (2019). Scrum Agile Framework in E-business Project Management: An Approach to Teaching Scrum. *European Project Management Journal*, *9*(1), 52–60. https://doi.org/10.18485/epmj.2019.9.1.7.

Misra, R., & Priyadarshini, R. (2018). Stakeholder Feedback System for Curriculum Design and Improvement – A case study. *Journal of Engineering Education Transformations*, *31*(Special Issue). https://doi.org/10.16920/jeet/2018/v0i0/120913.

Price, R., & Dunagan, A. (2019). Betting on Bootcamps: How Short-Course Training Programs Could Change the Landscape of Higher Ed. *Clayton Christensen Institute for Disruptive Innovation*, *April*.

Prihantoro, C. R. (2020). Vocational High School Readiness for Applying Curriculum Outcome Based Education (OBE) in Industrial 4.0 Era. *International Journal of Curriculum and Instruction*, *12*(1), 251–267.

Rawlley, A., & Mehra, C. (2020). Challenges in Implementation and Assessment of Outcome Based Education and Curriculum : Global Perspective. *JOURNAL OF ENGINEERING, SCIENCE & MANAGEMENT EDUCATION*, *13*(1), 17–23. https://doi.org/10.14710/ijee.1.2.122-129.

Saito, T., & Watanobe, Y. (2020). Learning Path Recommendation System for Programming Education Based on Neural Networks. *International Journal of Distance Education Technologies*, *18*(1), 36–64. https://doi.org/10.4018/IJDET.2020010103.

Salza, P., Musmarra, P., & Ferrucci, F. (2018). Agile Methodologies in Education: A Review. *Agile and Lean Concepts for Teaching and Learning*, 25–45. https://doi.org/10.1007/978-981-13-2751-3.

Scherer, R., Siddiq, F., & Viveros, B. S. (2020). A Meta-Analysis of Teaching and Learning Computer Programming: Effective Instructional Approaches and Conditions. *Computers in Human Behavior*, *109*. https://doi.org/10.1016/j.chb.2020.106349.

Shen, R., Wohn, D. Y., & Lee, M. J. (2019). Comparison of Learning Programming Between Interactive Computer Tutors and Human Teachers. *CompEd 2019 - Proceedings of the ACM Conference on Global Computing Education*, *16*, 48–58. https://doi.org/10.1145/3300115.3309506.

Stevanović, J., Atanasijević, S., & ... (2019). Raising the Skills of Business Analysts the Benefits of Elearning Technologies In Corporate Education. *In The 10th International Conference on e-Learning (eLearning-2019)*, 25–30. https://doi.org/10.13140/RG.2.2.28680.37129.

Steyn, C., Davies, C., & Sambo, A. (2018). Eliciting Student Feedback for Course Development: the Application of A Qualitative Course Evaluation Tool Among Business Research Students. *Assessment and Evaluation in Higher Education*, *44*(1), 11–24. https://doi.org/10.1080/02602938.2018.1466266.

Story, D., & Tait, A. (2019). Survey Research. *Anestheiosology*, *130*(2), 192–202.

Subih, M. A., Malik, B. H., Mazhar, I., Izaz-ul-Hassan, Sabir, U., Wakeel, T., Ali, W., Yousaf, A., Bilal-bin-Ijaz, Nawaz, H., & Suleman, M. (2019). Comparison of Agile Method and Scrum Method With Software Quality Affecting Factors. *International Journal of Advanced Computer Science and Applications*, *10*(5), 531–535. https://doi.org/10.14569/ijacsa.2019.0100569.

Tellhed, U., Björklund, F., & Strand, K. K. (2020). Sure I can code (but do I want to?). Why boys' and girls' programming beliefs differ and the effects of mandatory programming education. *Computers in Human Behavior*, *135*. https://doi.org/10.1016/j.chb.2022.107370.

Waguespack, L., Babb, J. S., & Yates, D. (2018). Triangulating Coding Bootcamps in IS Education: Bootleg Education or Disruptive Innovation? *Information Systems Education Journal*, *16*(6), 48–58.

Wu, S. (2021). Exploration and Practice of the Curriculum Reform of Real Estate Appraisal Based on OBE Education Concept. *ACM International Conference Proceeding Series*, 827–830. https://doi.org/10.1145/3456887.3457080.

Yang, J., Wong, G. K. W., & Dawes, C. (2018). An Exploratory Study on Learning Attitude in Computer Programming for the Twenty-First Century. *New Media for Educational Change*, 59–70. https://doi.org/10.1007/978-981-10-8896-4_5.

Younis, A. A., Sunderraman, R., Metzler, M., & Bourgeois, A. G. (2021). Developing parallel programming and soft skills: A project based learning approach. *Journal of Parallel and Distributed Computing*, *158*. https://doi.org/10.1016/j.jpdc.2021.07.015.

Zhan, Z., He, L., Tong, Y., Liang, X., & Lan, X. (2022). The effectiveness of gamification in programming education: Evidence from a meta-analysis. *Computers and Education: Artificial Intelligence*, *3*. https://doi.org/10.1016/j.caeai.2022.100096.

Zhang, J., Sziegat, H., Perris, K., & Zhou, C. (2019). More than Access: MOOCs and Changes in Chinese Higher Education. *Learning, Media and Technology*, *44*(2), 108–123. https://doi.org/10.1080/17439884.2019.1602541.