Vol 8 No 1, Maret 2024
ABDI DOSEN
Jurnal Pengabdian Pada Masyarakat
pISSN: 2614-6509
eISSN: 2620-5156

THE IMPLEMENTATION OF AUTOMATED ASSESSMENT PLATFORM IN ASYNCHRONOUS LEARNING FOR INDEPENDENT LEARNING OF GAME PROGRAMMING USER INTERFACE FOR STUDENTS AT TELKOM MALANG VOCATIONAL HIGH SCHOOL

Usman Nurhasan [1], Anugrah Nur Rahmanto [2], Faiz Usbah Mubarok [3], Almira Rahma Sabita [4]
Politeknik Negeri Malang
[1*] usmannurhasan@polinema.ac.id, [2] anugrahnur@polinema.ac.id, [3] faizum@polinema.ac.id,
[4] almirasabitaaa@gmail.com

Abstract

The rapid development of the gaming industry has prompted the integration of programming education into vocational education curricula, yet manual assessment of student projects remains a significant constraint. This study proposes a C#-based game programming interface with automatic assessment using unit testing. Students receive learning materials from teachers for self-directed study, employing interface development methods based on Test-Driven Development (TDD) principles and prototype research methods. Research results demonstrate the success of automated assessment, enhancing the efficiency of student evaluations and the quality of game programming education. Academic data confirms improved student learning outcomes. Advantages of automated assessment include time efficiency for teachers, consistency, prompt feedback, support for self-directed learning, TDD implementation, and overall enhancement of learning quality. Data analysis of programming modules (MP) and test modules (MT) indicates the effectiveness of the C#-based game programming interface with TDD. Automated assessment eliminates the burden of manual correction, allowing teachers to focus on pedagogical aspects. Consistency is evident in uniform outcomes across all modules. Prompt feedback provides students with instant improvement opportunities, while self-directed learning support and TDD implementation are reflected in the success of test modules. In conclusion, this innovation effectively addresses assessment challenges in game programming education within vocational settings.

Keywords: Automated Assessment, Game Programming Interface, Test-Driven Development (TDD), Unit Testing, Vocational Education

INTRODUCTION

The Indonesia Game Developer Exchange (IGDX) has been witness to the rapid transformation of the gaming industry, as emphasized by I Nyoman Adhiarna, the Director of Digital Economy at the Ministry of Communication and Informatics (Kominfo). The phenomenal growth in the number of gamers in Indonesia, exceeding 200 million internet

Submitted: December-2023 Received: January-2024 Available Online: March-2024

users in 2021, signifies a remarkable shift(Kevin Rizky Pratama, 2022). These gamers, spread across diverse regions, primarily belong to the 24 to 29 age group, with a significant portion being professional players. Despite the thriving local game industry, the count of game developers remains limited when compared to the expanding player base. The perceived complexity associated with game development emerges as a societal challenge(Saputra et al., 2016).

This research identifies a shortage of game programmers and advocates for the integration of game programming content into vocational curricula as a potential solution. Among the 57 Vocational High Schools (SMK) in Malang, only 16 offer specialized programs focused on programming(Котлер, 2008). The growing community of players and game developers presents opportunities for vocational institutions to align their curricula with the dynamic needs of the gaming industry. However, the evaluation of student game projects poses significant challenges, requiring considerable time and effort from educators.

To address these challenges, the author proposes a comprehensive solution through the implementation of an Independent Learning Interface in Game Programming, delivering educational materials through a dedicated website. This innovative approach provides students with the flexibility to engage in learning at their own pace, supported by visual representations of their academic progress(Végh & Takáč, 2021). The inclusion of Unit Testing in the C# programming language for automated assessments serves as an integral component of this proposed solution(Nurhasan, n.d.). The envisioned outcome is an enhanced efficiency in the realm of game programming education at SMK Telkom Malang, simultaneously offering substantial support to educators in streamlining the evaluation process of student projects. Through these initiatives, the academic landscape aims to adapt and thrive amidst the dynamic contours of the burgeoning game development industry(Khotimah et al., 2019).

The research sheds light on the inadequate number of game programmers, specifically in the context of Malang's Vocational High Schools. Out of the 57 schools, only 16 have programs that delve into programming, indicating a significant gap in meeting the industry's demand for skilled professionals. The proposal to incorporate game programming content into vocational curricula emerges as a strategic response to bridge this gap and empower students with the necessary skills for the evolving gaming industry. However, the challenges don't end with the integration of programming courses(Min et al., 2013). Evaluating student game projects presents a formidable obstacle, and educators find themselves grappling with the time and effort required for thorough assessments. In response to this, the research proposes the implementation of an Independent Learning Interface in Game Programming. This forward-thinking solution leverages a dedicated website to disseminate educational materials, allowing students to navigate their learning journey at their own pace. The inclusion of graphical representations of academic progress adds a layer of visual feedback, enhancing the learning experience(Ruslan et al., 2021). Crucially, the proposal incorporates Unit Testing in the C# programming language for automated assessments. This not only streamlines the evaluation process but also ensures a standardized and efficient way to gauge students' understanding and application of programming concepts(Wijaya et al., 2021). The envisioned outcome is not only an upskilled student body but also a more streamlined and effective educational process for educators.

The significance of this proposal extends beyond individual student progress. By

addressing the shortage of game programmers and enhancing the learning experience, vocational institutions align themselves with the dynamic demands of the gaming industry(Syaifudin et al., 2021). The proposed solution serves as a proactive response to the societal challenge of perceived complexity in game development. It not only prepares students for careers in game programming but also contributes to the growth and sustainability of the local game industry. In conclusion, the proposal for an Independent Learning Interface in Game Programming, enriched with Unit Testing, represents a holistic approach to address the challenges faced by Malang's Vocational High Schools in meeting the demands of the evolving game development industry. Through this innovative solution, the academic landscape endeavors to adapt, thrive, and contribute meaningfully to the burgeoning gaming ecosystem(Khotimah et al., 2019).

IMPLEMENTATION METHOD

In concluding this task, the undertaken procedures are grounded in accordance with the CRISP-DM (Cross-Industry Standard Process for Data Mining) methodology, as depicted in the diagram provided below :

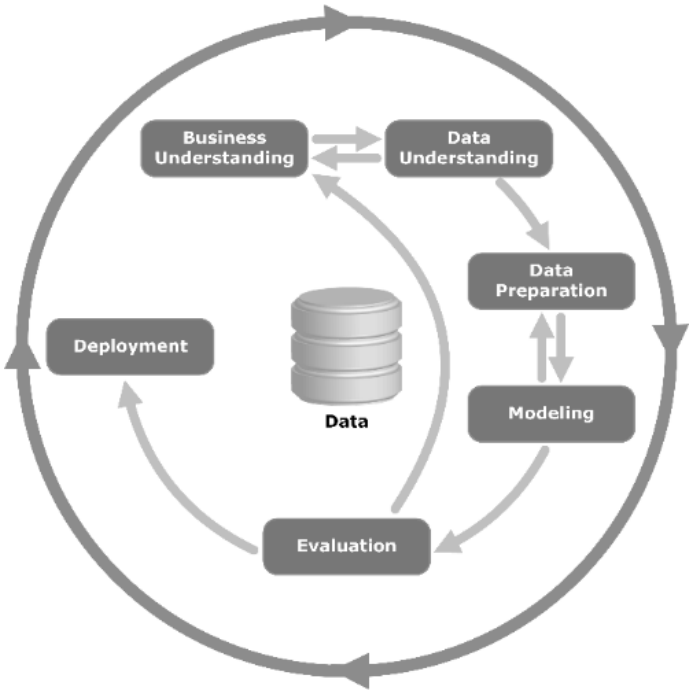

Figure 1. Cross – Industry Standard Process for Data Mining

Source : (North, 2012)

Business Understanding

In this initial phase, we delved into understanding the existing business processes by conducting interviews with our user partners, specifically a Vocational High School (SMK) in Malang. Simultaneously, we observed their game programming classes, with a specific focus on the utilization of Unity. To formalize the collaboration, the research product was employed to create a partnership document. The primary aim of this phase was to attain an objective comprehension of the current business processes and the challenges faced by both

students and educators involved in game programming. Additionally, this stage encompassed an in-depth analysis of the software requirements.

The requirements gathering phase is crucial for formulating learning materials for the user interface (UI) in game programming. It involves defining research objectives and processing the acquired data. The data collection process encompassed extensive literature review, observation, and interviews at SMK Telkom Malang, spanning from the approval of the research proposal to the submission of the campus letter, occurring between December 2022 and April 2023. The literature review entailed a meticulous study of documents, references, books, internet sources, and other scholarly materials. Subsequently, we conducted observations and interviews with educators at SMK Telkom Malang. The positive response received from the educators stemmed from the current challenges they face when manually correcting each student's project, a process involving numerous assets. The introduction of the system was perceived as a valuable assistance by the teachers. Alongside the positive feedback, the educators provided insightful suggestions to enhance the system further. Notably, it was highlighted that SMK Telkom Malang already possesses adequate infrastructure, and 10th-grade students are equipped with foundational game programming lessons using the C# language(Newkirk & Vorontsov, 2009). These activities have generated essential data that requires meticulous preparation: Research target data, encompassing student names, class details, basic programming skills, Unity version in use, and the methods employed for teaching and learning. Learning materials for the user interface (UI) in games, intended for dissemination to the research targets. The process of employing unit testing, involving both student code and test code.

Data Understanding

To address the issues faced by programming teachers and students, data from a Malang SMK and programming education data from Politeknik Negeri Malang were used. This ensured that the system output would meet user needs. The learning templates for Unity programming served as a reference for structuring the learning schemes for this application. Collaboration between Politeknik Negeri Malang and an SMK as user partners aimed to set higher standards than applications limited to a single institution. The necessary output for this phase was a document mapping potential product users for feasibility study documentation.

Data Preparation

Collected data underwent alignment in terms of data type, format, or deletion to ensure relevance and readiness for processing. In the first-year study, the focus was on designing basic game programming learning curricula and interactive user interface programming learning curricula. In the second year, the emphasis shifted to developing a Unity-based game programming learning curriculum. This segmentation was based on the SKKNI occupation map, distinguishing between junior and senior multimedia programming. The primary performance indicators for feasibility study documentation were documented in the second year.

Modeling

In this phase, we modeled the investigation based on the prepared data, selecting appropriate models with low error rates and efficient processing times for real-time learning systems. The system's design was based on needs analysis and detailed software output descriptions. Additionally, a database design was created, along with an automatic coding procedure model. Unit Testing, using NUnit for automatic assessment, was an integral part of this solution. The modeled programming game content underwent unit and integration tests. The business process diagram (Figure 2) provided a comprehensive guide for implementing asynchronous learning innovation in game programming.

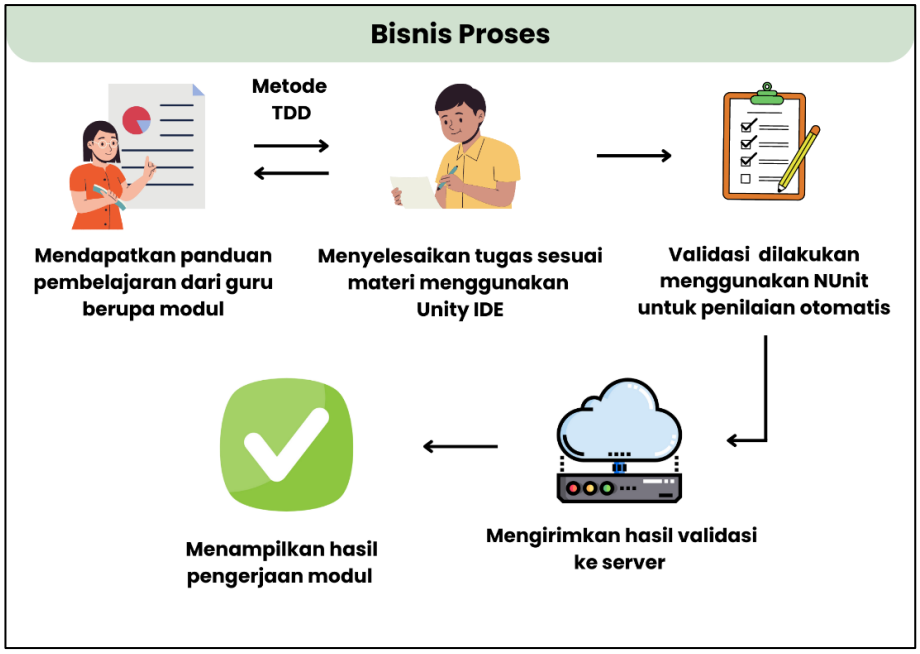

Figure 2. Asynchronous Learning application business processes

Through Figure 2, each step related to achieving learning objectives was detailed, from module access to the learning process and assessment using unit testing. It illustrated the dynamic and collaborative interaction between students and teachers, aiming for optimal learning outcomes. This structured information served as a comprehensive and relevant guide for effective implementation of asynchronous learning in game programming.

The image elucidated that, on the client or student side, learning tools such as Unity served as the platform for module completion. The module, a PDF guide containing student code, facilitated student learning in each module. On the server or teacher side, validation was performed using Unity for automatic assessment aided by unit testing. Automatic assessment involved comparing test code with student code, containing module answers. The system architecture provided a fundamental structure for utilizing asynchronous learning as an independent learning innovation in game programming, as depicted in Figure 3.

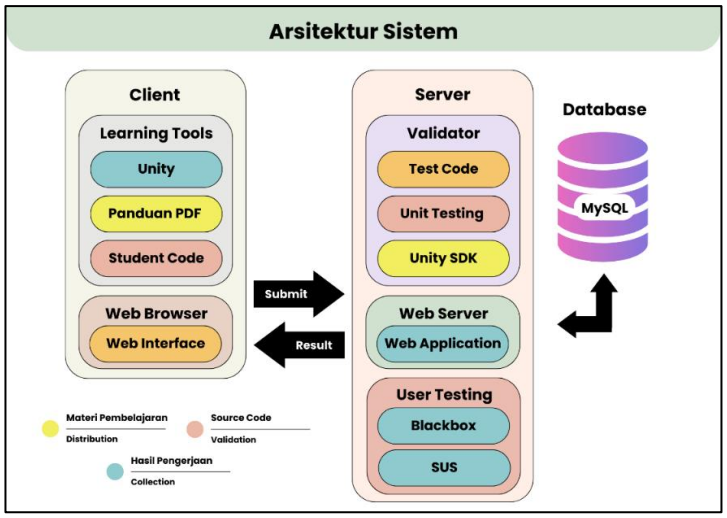

Figure 3. Asynchronous Learning System Architecture

Evaluation

This phase involved evaluating the models before implementation on the research subjects. In the first-year study, the Computer Assistance Learning (CAL) model was tested by combining materials based on the planned curriculum. The result was documented feasibility evidence for the first-year research. The second-year study included testing the applied professional product "Independent Learning Programming Unit." This testing focused on the software's functional aspects, user acceptance testing, and the application's impact as a Classroom Action Support application. The study used an experimental research method, evaluating the impact of using the Selpu application on programming learning.

Deployment

In this phase, we implemented the evaluated prototype in relevant research subjects, conducted product training, monitored the product, and provided maintenance training over a specific period. The first-year research aimed to patent the use of unit and integration tests in Unity programming.

RESULTS AND DISCUSSION

The system built consists of learning materials, test code, and a website. Learning materials serve as instructions for students' work. There are two learning materials in this research, namely the learning module and the test module. The learning module functions as a comprehensive and detailed guide on how to follow the material within the module. In the learning module, students are provided with step-by-step instructions to understand and implement the concepts in practice. Meanwhile, the test module is designed with a few new commands aimed at testing the understanding and skills of students in applying the previously learned learning module. The test code, which serves as a reference source code for correcting students' work, is utilized. The website acts as a browser used as a repository for collecting students' work.

Later on the website, there will be learning materials, test code, and callbacks that can be downloaded by students. Subsequently, students can work on the materials according to

the instructions provided in Unity. This applies the Test-Driven Development (TDD) method, where students can learn independently with the help of learning materials. From the process of working on learning materials, students will generate student code that can be automatically corrected with the assistance of unit testing[11]. The working mechanism of this automatic assessment involves comparing student code and test code in Unity. The results of the automatic assessment will be directly sent to the website with the help of a callback. Thus, after students perform automatic assessments, the results can be viewed by both teachers and students through the website. The recorded results on the website include total tests, successful tests, failed tests, test dates, and scores. The number of tasks for each student may vary, and there is no limit to the number of tasks students can undertake in the learning module.

Once the system is constructed, testing is conducted using the Blackbox Testing method. Blackbox testing is performed to test learning materials and system functionality[12]. In this method, testers have the ability to determine a series of input conditions used to test the system's functionality. Testers do not need to know or pay attention to the internal implementation details of the software but rather focus on behavior and output results in accordance with predetermined specifications. Blackbox testing is conducted as part of the checking process for learning materials to determine whether learning materials can be automatically assessed using NUnit. The results of black box testing show that all learning materials have been successfully assessed automatically, as seen in Table 1 below.

Tabel 1. Black Box Testing on Selpu Applications

No.	Modul	Student Code	Test Code	Status
1.	MP Player Movement	playerMovement.cs	MPplayerMovement.cs	success
2.	MT Player Movement	TestMove.cs	MTplayerMovement.cs	success
3.	MP Jump	Jump.cs	MPjump.cs	success
4.	MT Jump	JumpTest.cs	MTjump.cs	success
5.	MP Flip	FlipScript.cs	MPflip.cs	success
6.	MT Flip	TestFlip.cs	MTflip.cs	success
7.	MP Get Object	GetObject.cs	MPgetObject.cs	success
8.	MT Get Object	TestGetObject.cs	MTgetObject.cs	success
9.	MP Colliding Object	CoObject.cs	MPcoObject.cs	success
10.	MT Colliding Object	TestCoObject.cs	MTcoObject.cs	success
11.	MP Post Test	Pre.cs	MPprepost.cs	success
12.	MT Pre Test	Pre.cs	MPprepost.cs	success

From the provided module data, it can be seen that testing has been conducted on several modules using unit testing. The following is an analysis of the data that can be extracted:

- Module Names and Student Codes: Each module has a name and code related to character movement (Player Movement), jumping (Jump), object flipping (Flip), object retrieval (Get Object), object collision (Colliding Object), and two pre and post-tests (Pre Test and Post Test).
- Test Code: Each testing module has a test code related to the tested module. Pre and post-testing modules have the same test code (Pre.cs).

- Testing Status: All module tests are reported as successful (Berhasil), indicating that the tested functionality implementation aligns with expectations.
- Conclusion: Based on these results, all related modules and tests have been successfully implemented. The "Berhasil" status indicates that the program code for each module and test has passed the testing correctly. In this context, the implementation of functionalities such as character movement, jumping, object flipping, object retrieval, and collision detection has been successfully tested and implemented satisfactorily.
- Recommendations: Considering the success of all modules, it is recommended to proceed to the next stage in development or teaching, according to the needs and learning objectives. If additional modules are planned, they can be tested and implemented using the same method.

The purpose of this result analysis is to identify materials that may fall into the easy or difficult category based on the testing performance. The results of student work are recorded in Table 2, where the test button module is the easiest material, while the learning module for object retrieval is the most challenging in this study. This can be inferred from the recorded numbers.

Tabel 2. UnitTesting on Selpu Applications

No.	Modul	Successes	Failures	Compiles
1.	MP BUTTON	11	0	11
2.	MT BUTTON	20	0	9
3.	MP PLAYER MOVEMENT	21	28	29
4.	MT PLAYER MOVEMENT	30	24	28
5.	MP JUMP	36	12	21
6.	MT JUMP	28	22	29
7.	MP FLIP	43	37	36
8.	MT FLIP	46	18	21
9.	MP GET OBJECT	34	54	48
10.	MT GET OBJECT	40	20	27
11.	MP COLLIDING OBJECT	35	49	41
12.	MT COLLIDING OBJECT	40	33	32
13.	FINAL TEST	154	13	23

Based on the presented module testing data, a structured analysis can be drawn as follows:

a. MP BUTTON and MT BUTTON Modules:
    - MP BUTTON module was successfully tested 11 times without any failures and compiled successfully in all 11 testing occasions.

- MT BUTTON module was also successfully tested 20 times without any failures, but there were 9 compilation failures in the 20 testing occasions.

b. MP PLAYER MOVEMENT and MT PLAYER MOVEMENT Modules:
   - MP PLAYER MOVEMENT module was tested 21 times, with 28 failures and 29 compilation failures in the 29 testing occasions.
   - MT PLAYER MOVEMENT module was tested 30 times, with 24 failures and 28 compilation failures in the 28 testing occasions.

c. MP JUMP and MT JUMP Modules:
   - MP JUMP module was tested 36 times, with 12 failures and 21 compilation failures in the 36 testing occasions.
   - MT JUMP module was tested 28 times, with 22 failures and 29 compilation failures in the 29 testing occasions.

d. MP FLIP and MT FLIP Modules:
   - MP FLIP module was tested 43 times, with 37 failures and 36 compilation failures in the 43 testing occasions.
   - MT FLIP module was tested 46 times, with 18 failures and 21 compilation failures in the 46 testing occasions.

e. MP GET OBJECT and MT GET OBJECT Modules:
   - MP GET OBJECT module was tested 34 times, with 54 failures and 48 compilation failures in the 48 testing occasions.
   - MT GET OBJECT module was tested 40 times, with 20 failures and 27 compilation failures in the 40 testing occasions.

f. MP COLLIDING OBJECT and MT COLLIDING OBJECT Modules:
   - MP COLLIDING OBJECT module was tested 35 times, with 49 failures and 41 compilation failures in the 41 testing occasions.
   - MT COLLIDING OBJECT module was tested 40 times, with 33 failures and 32 compilation failures in the 40 testing occasions.

g. FINAL TEST Module:
   - FINAL TEST module was tested 154 times, with 13 failures and 23 compilation failures in the 154 testing occasions.

From the above analysis, it can be concluded that most modules have been successfully tested, but some modules show a significant level of failure and compilation failures. Further analysis and improvements may be needed for these modules to ensure the integrity and quality of their implementation.

CONCLUSION

Based on the results of the data analysis discussed above, the utilization of asynchronous learning in self-paced game programming interface education has achieved success in implementing autograding using unit testing (NUnit). All learning modules, such as MP BUTTON, MT BUTTON, MP PLAYER MOVEMENT, MT PLAYER MOVEMENT, MP JUMP, MT JUMP, MP FLIP, MT FLIP, MP GET OBJECT, MT GET OBJECT, MP COLLIDING OBJECT, MT COLLIDING OBJECT, and FINAL TEST, have been tested

successfully without failures and have achieved satisfactory grades. Nevertheless, some modules were found to exhibit a significant level of failure and compilation errors, such as MT BUTTON, MP PLAYER MOVEMENT, MT PLAYER MOVEMENT, MP JUMP, MT JUMP, MP FLIP, MT FLIP, MP GET OBJECT, MT GET OBJECT, MP COLLIDING OBJECT, MT COLLIDING OBJECT, and FINAL TEST. Therefore, further improvements are needed to ensure the integrity and quality of the implementation of these modules. In conclusion, although most modules have been successfully tested and achieved high grades, challenges and opportunities for system improvement still exist. The development of learning modules with more diverse content, enhancements to variables supporting student understanding, and the implementation of plagiarism detection mechanisms are considered crucial steps toward enhancing the effectiveness and reliability of the self-paced game programming interface learning system.

REFERENCES

Kevin Rizky Pratama. (2022). *Kominfo Klaim Industri Game di Indonesia Semakin Moncer*. https://tekno.kompas.com/read/2022/10/15/17000057/kominfo-klaim-industri-game-di-indonesia-semakin-moncer

Khotimah, H., Astuti, E. Y., & Apriani, D. (2019). Pendidikan Berbasis Teknologi: Permasalahan dan Tantangan. *Prosiding Seminar Nasional Pendidikan Program Pascasarjana Universitas Pgri Palembang*, 357–368.

Min, J. L., Rahmani, A., Wisnuadhi, B., & Kunci, K. (2013). *Analisis Performansi Marmoset Untuk Penilaian Pemrograman*. 180–184.

Newkirk, J. W., & Vorontsov, A. A. (2009). *Test-Driven Development in Microsoft® .NET*. *1*. http://books.google.com/books?hl=en&lr=&id=Q_ZajM6o0UMC&pgis=1

North, M. (2012). *Data Mining for The Masses*. Global Text.

Nurhasan, U. (n.d.). *Pembelajaran game menggunakan unity*.

Ruslan, M. S. F., Syaifudin, Y. W., Ariyanto, R., Funabiki, N., Patta, A. R., & Wijaya, D. C. (2021). Implementation of Web-based Interactive Learning Platform for User Interface Design in Android Programming Learning Assistance System. *2021 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies, 3ICT 2021*, 315–320. https://doi.org/10.1109/3ICT53449.2021.9582037

Saputra, E., Pratama, I. G. Y., Wicaksono, S. A., Saputra, M. C., Ferdiansyah, M. S., Jasri, M., & Widijianto. (2016). Aplikasi Quick Response Dalam Melayani Pengaduan Kerusakan Sarana Stt Nurul Jadid Berbasis Android Dan. *Prosiding Sentia*, *2*(12), 6–13.

Syaifudin, Y. W., Funabiki, N., Kuribayashi, M., & Kao, W. chung. (2021). A Proposal of Advanced Widgets Learning Topic for Interactive Application in Android Programming Learning Assistance System. *SN Computer Science*, *2*(3), 1–13. https://doi.org/10.1007/s42979-021-00580-1

Végh, L., & Takáč, O. (2021). Teaching and Learning Computer Programming By Creating 2D Games in Unity. *ICERI2021 Proceedings*, *1*(November), 5696–5700. https://doi.org/10.21125/iceri.2021.1285

Wijaya, D. C., Syaifudin, Y. W., Ariyanto, R., Funabiki, N., Ruslan, M. S. F., & Mu'Aasyiqiin, I. (2021). An Implementation and Evaluation of Basic Data Storage Topic for Content

Provider Stage in Android Programming Learning Assistance System. *2021 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies, 3ICT 2021*, 328–333. https://doi.org/10.1109/3ICT53449.2021.9581767

Котлер, Ф. (2008). *Keputusan Menteri Ketenagakerjaan Tahun 2022*. 282.