Online version at https://journal.lenterailmu.com/index.php/jafotik

JAFOTIK

E-ISSN: 3031-2698, DOI prefix: 10.70356

Development of a Student Code of Conduct Information System Using the Laravel and Agile Methodology at SMKN 1 Bonjol

Rahmi Fadila [1], Ami Anggraini Samudra [2,*], Herisvan Hendra [3]

[1,2,3] *Fakultas Sains dan Teknologi, Universitas PGRI Sumatera Barat, Padang, Indonesia*

ARTICLE INFO

Article history:
Received August 2026
Accepted 22 August 2026
Published 25 August 2026

Keywords:
Information System
Student Discipline
Laravel Framework
Agile Development Method
SMKN 1 Bonjol

ABSTRACT

Managing student discipline at SMKN 1 Bonjol was previously conducted manually using logbooks, which made the process prone to data loss, duplication, and delays in communicating student coaching information to parents. This study aims to design and develop a web-based Student Discipline Information System (SIMTATIB). The research employed a Research and Development (R&D) methodology using the Agile System Development Life Cycle (SDLC), consisting of the Requirement, Design, Development, Testing, Deployment, and Review stages. The system was developed using the Laravel 13 framework, PHP 8.4, and MySQL, and integrated with the WhatsApp Gateway via Fonnte to enable automated notifications. Alpha testing using Whitebox Testing produced acceptable Cyclomatic Complexity values, while Blackbox Testing achieved a 100% success rate. Beta testing based on the ISO/IEC 25010 standard, conducted by system experts, achieved an average score of 96.24%, while evaluation by Counseling Guidance (BK) teachers demonstrated that the system is highly feasible and practical for optimizing student discipline management.

1. Introduction

Information and communication technology (ICT) has advanced rapidly, driving digital transformation in education and enabling schools to manage administrative processes more efficiently, accurately, transparently, and in an integrated manner [1]. In vocational high schools (Sekolah Menengah Kejuruan/SMK), particularly SMKN 1 Bonjol, the enforcement of school rules and student discipline is an essential component of character development and mental preparedness before students enter the workforce and industrial environment. However, the monitoring and documentation of student discipline still face various operational challenges, particularly due to the continued use of traditional record-keeping systems [2]. Based on initial observations and data analysis at SMKN 1 Bonjol, daily student violations are still recorded manually using paper-based recapitulation books. The recorded daily violations consist of approximately 60% minor violations, such as tardiness and incomplete school uniforms or attributes, 34% moderate violations, and 5% serious violations.

* *Corresponding author:* Ami Anggraini Samudra
E-mail address: amianggrainisamudra@gmail.com

Manual data management creates several fundamental problems, including an increased risk of lost or damaged records, potential duplication of student data, and lengthy recapitulation processes when Counseling Guidance (BK) teachers and the student affairs team require historical discipline records. To address these challenges, a web-based Student Discipline Information System (SIMTATIB) was designed and developed. Unlike conventional student discipline systems that typically rely on point-based scoring, the proposed system employs an action-oriented digital logbook based on a checklist approach. This method focuses on documenting the frequency of violations and directly determining the appropriate stage of disciplinary guidance without relying on numerical scores that may be difficult to interpret. The system was developed using the Laravel 13 framework, PHP 8.4, and a MySQL database [3]. Laravel was selected because of its well-structured Model-View-Controller (MVC) architecture, built-in security features such as Cross-Site Request Forgery (CSRF) protection and password hashing, and ease of application maintenance [4]. The system development process adopted the Agile Software Development Life Cycle (SDLC) methodology. The Agile approach enables software development to be conducted incrementally, adaptively, and responsively to the needs and feedback of school stakeholders [5]. Through the development of SIMTATIB, this study is expected to provide an effective, accountable, and efficient digital solution for supporting student discipline management and character development at SMKN 1 Bonjol.

2. Method

Figure 1 illustrates the research methodology employed to achieve the objectives established at the beginning of the study. Agile Development is an iterative and incremental software development approach that emphasizes flexibility, collaboration, continuous feedback, and adaptation to changing requirements [6]–[13]. Agile enables development teams to deliver software in small increments while continuously evaluating and improving the system based on user and stakeholder feedback [6], [7]. Common Agile approaches include Scrum, Extreme Programming, and other adaptive practices that support effective project management and software delivery [8], [9]. Studies have shown that Agile can improve communication, customer involvement, and responsiveness to changing requirements [10], [11]. Its effectiveness, however, depends on the selected practices and the organizational context in which Agile is implemented [12], [13]. The system development process adopted the Agile model, an iterative approach to software development that enables continuous refinement throughout the development process. The Agile methodology consists of six stages: Requirement, Design, Development, Testing, Deployment, and Review [14].

2.1 Requirement Analysis

This stage focuses on identifying the functional and non-functional requirements of the system. The functional requirements include role-based access management, violation category management, violation recording, disciplinary action recording, and report generation. The non-functional requirements cover the hardware and software requirements, as well as the specifications of the hosting server.

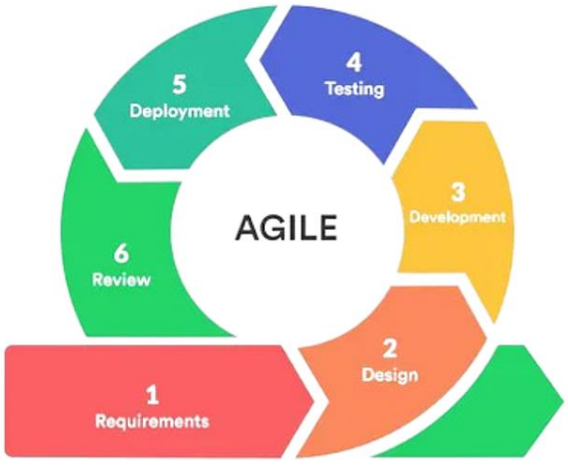

Fig. 1 - Agile methodology

2.2 Design

At this stage, the system architecture is modeled using the Unified Modeling Language (UML), which includes the Use Case Diagram, Activity Diagram, Class Diagram, and Sequence Diagram. In addition, the relational database schema is designed to support the system's data management requirements.

A. Use Case Diagram

A use case is a modeling technique used to describe a system's requirements and define the functions that the system is expected to provide. The main components of a use case are actors, use cases, and the subject (system) [15]. The Use Case Diagram illustrates the main functions and interactions between users and the Student Discipline Information System (SIMTATIB), as presented in Figure 2.

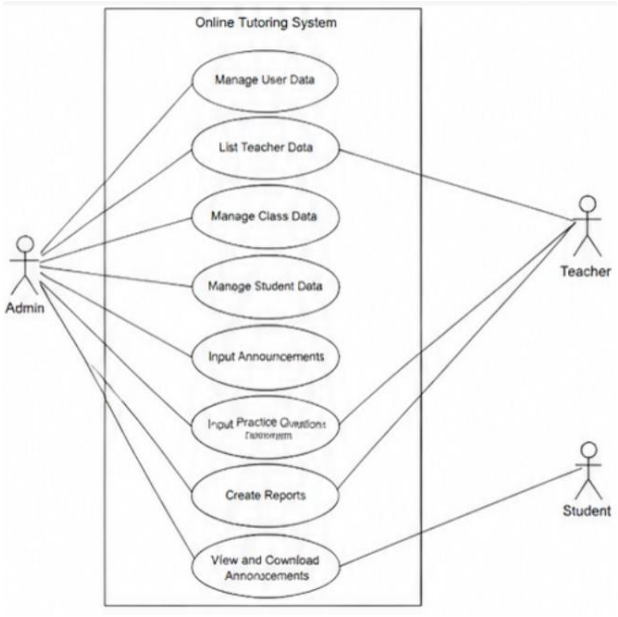

Fig. 2 – Use Case Diagram

B. Activity Diagram

An activity diagram illustrates the workflow or activities of a system or another process. Figure 3 presents an example of a portion of the proposed Student Discipline Information System (SIMTATIB) activity diagram.

C. Class Diagram

A class diagram illustrates the structure of a system in terms of defining the classes that will be created to build the system. The class diagram in Figure 4 shows the classes in the Student Discipline Information System (SIMTATIB).

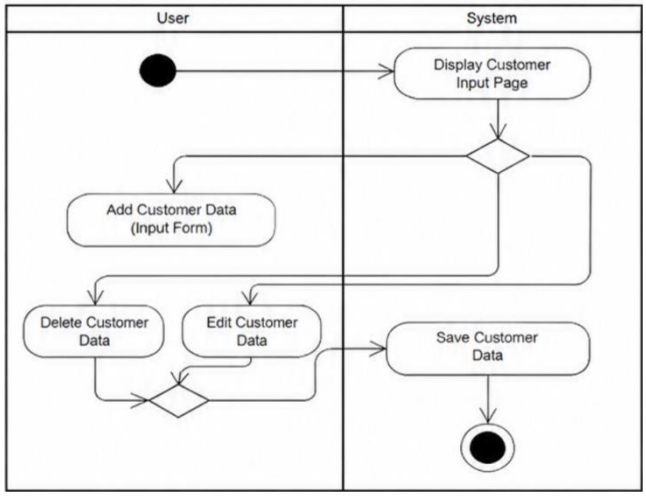

Fig. 3 – Activity Diagram

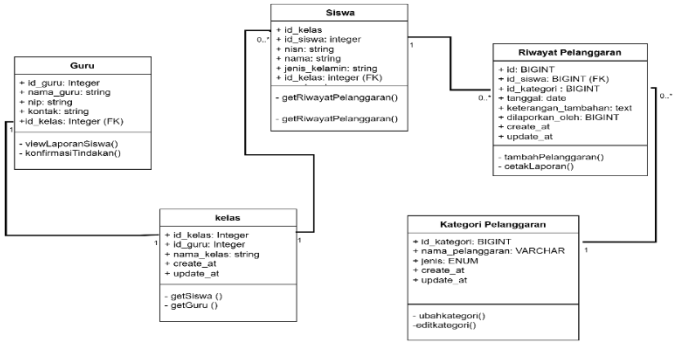

Fig. 4 – Class Diagram

D. Sequence Diagram

The sequence diagram in Figure 5 illustrates how an operation is performed and displays the messages sent as a result of interactions between objects in the Student Discipline Information System (SIMTATIB).

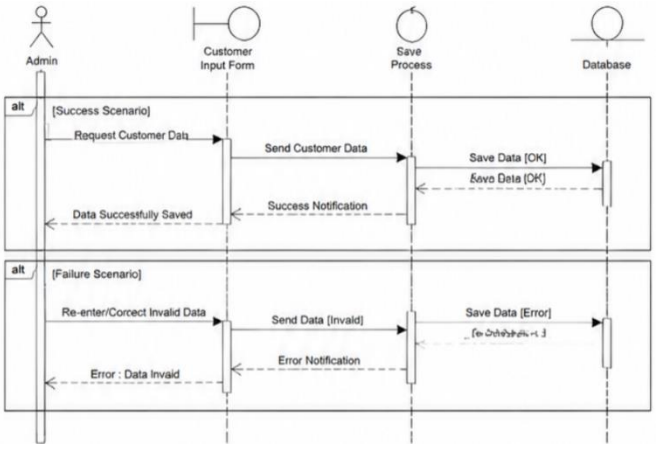

Fig. 5 – Sequence Diagram

2.3 Development

In this development stage, the system programmer begins writing the program code according to the system design that has been developed and the functional requirements that have been identified. The program is implemented using PHP 8.4, Laravel 13, and MySQL as the database. The code is written using Visual Studio Code with Laragon as the local server environment.

2.4 Testing

Alpha testing is conducted by the developers and expert lecturers using White-box Testing and Black-box Testing to evaluate all button functionalities and interface navigation. Beta testing is conducted to evaluate the quality of the system by two lecturers and one teacher based on the ISO/IEC 25010 standard.

2.5 Deployment

This stage transforms the system from initially being accessible only locally into a system that can be accessed through the internet. The application is deployed to a hosting-based production environment (Arenhost) so that it can be accessed by the school in real time.

2.6 Review and Evaluation

The results of this stage are used to identify new features that emerge during the design process, plan further development, and make the necessary improvements to enhance the quality of the Student Discipline Information System (SIMTATIB) that has been developed.

3. Result and Discussion

Figure 6 presents the developed login page of the Student Discipline Information System (SIMTATIB). Administrators and homeroom teachers log in using their email addresses, while students log in using their respective NISN (National Student Identification Numbers).

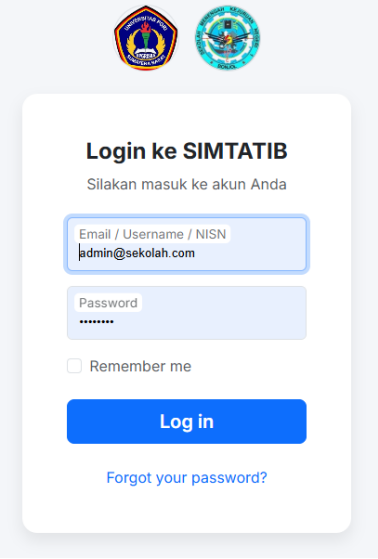

Fig. 6 – Login Form

Figure 7 presents the developed dashboard page of the Student Discipline Information System (SIMTATIB). This page provides administrators with full access to system features, allowing them to manage student and homeroom teacher data, record student violations, and generate and print reports.

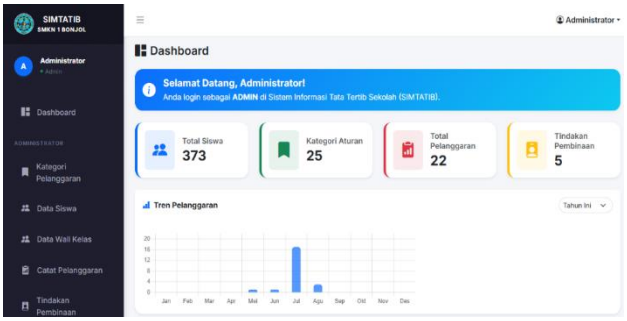

Fig. 7 - Admin Dashboard Development Results

Figure 8 presents the developed violation recording page of the Student Discipline Information System (SIMTATIB). On this page, administrators can record violations committed by students.

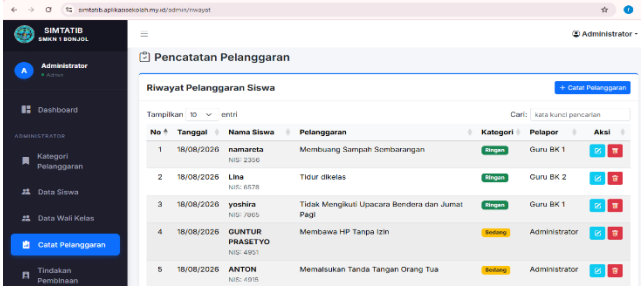

Fig. 8 - Violation Recording Page Development Results

3.1. Whitebox and Blackbox Testing

The purpose of testing is to verify whether the software's functionality, inputs, and outputs meet the specified requirements [16]. White-box testing is conducted to analyze the internal logical structure of the program across the system's main workflows, including login, violation categories, student data, homeroom teacher data, and violation recording. In this testing process, the tester focuses solely on the system's inputs and outputs to ensure that each feature operates according to the specified requirements and system specifications. Black-box testing focuses on verifying interface functionality and the system's response to user inputs. The testing covers the main modules and scenarios, including login, data management, violation recording, and report generation. The test results show that 100% of the test cases were successfully executed without any errors.

3.2. Expert Evaluation

This testing focuses on assessing the extent to which the quality of the developed software meets expectations or still requires improvement, as well as evaluating the system's suitability based on specific criteria. Beta testing was conducted by three system experts and covered the eight software quality characteristics defined in ISO/IEC 25010. The results of the assessment are summarized in the following Table 1:

Table 1 - Expert Testing Results

Criteria	Score Percentage (100%)	Description
Functionality	97.77%	Highly Feasible
Usability	95.55%	Highly Feasible
Security	96.66%	Highly Feasible
Performance Efficiency	93.33%	Highly Feasible
Compatibility	100%	Highly Feasible
Reliability	93.33%	Highly Feasible
Maintainability	100%	Highly Feasible
Portability	93.33%	Highly Feasible
Average	96.24%	Highly Feasible

With an average score of 96.24%, the system is categorized as "Highly Feasible" and meets the technical software suitability standards for use in a school environment.

3.3. Users Evaluation

User testing was conducted with respondents consisting of Guidance and Counseling (BK) teachers. The results of the beta testing based on user assessments are presented in the following Table 2:

Table 2 – Users Evaluation Results

Criteria	Score Percentage (100%)	Description
Content	92.5%	Highly Feasible
Accuracy	95%	Highly Feasible
Format	98%	Highly Feasible
Ease of Use	97.5%	Highly Feasible
Timeliness	97.5%	Highly Feasible
Average	96.1%	Highly Feasible

Based on the table above, the user assessment achieved an average score of 96.1%, which falls into the "Highly Feasible" category.

4. Conclusion

This study successfully developed a Student Discipline Information System using the Laravel 13 framework with an Agile Development approach at SMKN 1 Bonjol. The system replaces the manual recording process with an efficient and transparent action-oriented digital logbook. Alpha testing, consisting of White-box and Black-box Testing, demonstrated that the system's internal logic and functionality were 100% valid. Meanwhile, beta testing based on the ISO/IEC 25010 standard achieved an average score of 96.24%, categorized as "Highly Feasible." Several recommendations can be considered for future research and further development. The system's features could be expanded by incorporating student character analysis, making the system more adaptive and feature-rich.

Acknowledgements

The authors would like to express their gratitude to the school leadership, the Vice Principal for Student Affairs, and the Guidance and Counseling (BK) teachers of SMKN 1 Bonjol for their permission, support, facilities, and cooperation throughout the research and system testing process. The authors would also like to thank their supervisors for their academic guidance, valuable insights, and support, which greatly contributed to the successful completion of this research.

REFERENCES

[1] Okpatrioka, "Okpatrioka STKIP Arrahmaniyah," DHARMA ACARIYA NUSANTARA : Jurnal Pendidikan, Bahasa dan Budaya, 2023.

[2] D. Okta Putra, "Penerapan Metode Agile Pada Aplikasi E-Point Pelanggaran Tata Tertib Siswa," 2023.

[3] A. Sultan Adensa, K. Raihan, R. Faisal Rafi, I. Richwandi Putra, and F. Azizah, "Pengembangan Web Dinas Perpustakaan dan Arsip Berbasis Laravel Framework pada DPAD Kota Tangerang," 2023, doi: https://doi.org/10.36040/jati.v7i6.7840.

[4] R. Yuniarti, I. H. Santi, and W. D. Puspitasari, "Perancangan Aplikasi Point Of Sale Untuk Manajemen Pemesanan Bahan Pangan Berbasis Framework Laravel," 2022, doi: https://doi.org/10.36040/jati.v6i1.4283.

[5] N. H. Ade Maulana, Rekayasa PerangkatLunak: Konsep, Metpde, dan Praktik Terbaik. 2023.

[6] T. Dybå and T. Dingsøyr, "Empirical studies of agile software development: A systematic review," Information and Software Technology, vol. 50, no. 9–10, pp. 833–859, 2008, doi: https://doi.org/10.1016/j.infsof.2008.01.006.

[7] P. Abrahamsson, O. Salo, J. Ronkainen, and J. Warsta, Agile Software Development Methods: Review and Analysis. Espoo, Finland: VTT Publications, 2002.

[8] P. Abrahamsson, J. Warsta, M. T. Siponen, and J. Ronkainen, "New directions on agile methods: A comparative analysis," in Proc. 25th Int. Conf. Software Engineering (ICSE), 2003, pp. 244–254, doi: https://doi.org/10.1109/ICSE.2003.1201204.

[9] F. Paetsch, A. Eberlein, and F. Maurer, "Requirements engineering and agile software development," in Proc. 12th IEEE Int. Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE), 2003, pp. 308–313, doi: https://doi.org/10.1109/ENABL.2003.1231428.

[10] M. Pikkarainen, M. Haikara, O. Salo, P. Abrahamsson, and J. Still, "The impact of agile practices on communication in software development," Empirical Software Engineering, vol. 13, pp. 303–337, 2008, doi: https://doi.org/10.1007/s10664-008-9065-9.

[11] A. Sampaio, A. Vasconcelos, and P. R. Falcone Sampaio, "Assessing agile methods: An empirical study," Journal of the Brazilian Computer Society, vol. 10, pp. 21–48, 2004, doi: https://doi.org/10.1007/BF03192357.

[12] W. N. Behutiye, P. Rodríguez, M. Oivo, and A. Tosun, "Analyzing the concept of technical debt in the context of agile software development: A systematic literature review," Information and Software Technology, vol. 82, pp. 139–158, 2017, doi: https://doi.org/10.1016/j.infsof.2016.10.004.

[13] M. Kuhrmann et al., "What makes agile software development agile?" IEEE Transactions on Software Engineering, vol. 48, no. 9, pp. 3523–3539, 2022, doi: https://doi.org/10.1109/TSE.2021.3099532.

[14] W. D. Prastowo, D. Danianti, and A. Pramuntadi, "Analisis Risiko Pada Pengembangan Perangkat Lunak Menggunakan Metode Agile Dan RAD (Rapid Application Development)," Citizen : Jurnal Ilmiah Multidisiplin Indonesia, vol. 3, no. 3, pp. 169–174, Aug. 2023, doi: https://doi.org/10.53866/jimi.v3i3.388.

[15] Shalahuddin. M. A.S Rosa, Rekayasa Perangkat Lunak (Terstruktur dan Berorientasi Objek). 2016.

[16] F. Asfiana Putri, G. Indah Marthasari, and I. Nuryasin, "Rancang Bangun Perangkat Lunak Perhitungan Metrik Cyclomatic Complexity Berdasarkan Control Flow Graph Berbasis Web," REPOSITOR, vol. 5, no. 1, pp. 565–574, 2023, doi: https://doi.org/10.22219/repositor.v5i1.31118.