Published online on the page: https://jurnal.ilcagam.org/index.php/jsti

J S T I
Jurnal Serumpun Teknik Informatika
| ISSN (*Online*) 3123-965x | DOI: https://doi.org/10.66485/jsti.v1i1.22

RESTful API Development for Student Schedule and Attendance Management in a Higher Education Environment

Rizky Parlika[1,], Abidin Sulaiman[2], Riky Hermawan[3]*

[1,3]Universitas Pembangunan Nasional "Veteran" Jawa Timur, Surabaya, Indonesia
[2]Akademi Manajemen Informatika dan Komputer Kosgoro, Solok, Indonesia

Article Information

Article History:
Submit: 26 Maret 2026
Accepted: 25 April 2026
Published: 30 April 2026

Keywords

RESTful API; Student Attendance; Class Schedule; Laravel; Academic Information System.

Correspondence

E-mail: abidinsulaiman@amikkosgoro.ac.id*

ABSTRACT

This study presents the development of a RESTful API service to support student schedule and attendance management in a higher education environment. The research is motivated by the fact that schedule management and attendance recording are often still handled manually or by stand-alone applications, which complicates recap processes, attendance monitoring, and integration with existing academic information systems. The proposed system is implemented using the Laravel framework and MySQL database, where student, lecturer, course, schedule, and attendance entities are modeled in a structured way and exposed through RESTful endpoints over HTTP with JSON data format. The research adopts a software engineering approach consisting of requirement analysis, system design, implementation, and testing using Postman on a local development environment. The experimental results show that all CRUD operations and attendance recording functions work as expected, producing consistent JSON responses with appropriate HTTP status codes, indicating that the developed API is suitable to be used as a foundation for future integration with web and mobile applications.

1. Introduction

Higher education institutions require an integrated information system to effectively manage academic processes, including scheduling classes and tracking student attendance (Surakarta et al., 2018)(Dan et al., 2022). In many academic programs, these activities are still carried out manually using paper attendance sheets or separate applications, as reported in various studies on attendance systems that still rely on signatures or manual input, thereby complicating the process of compiling attendance records, monitoring student discipline, and integrating with other academic information systems (Hamdani et al., 2024)(Syifa et al., 2025). This situation has the potential to lead to duplicate entries, delays in information, and a high risk of data entry errors (Qr-code et al., 2021).

As service-oriented software architecture continues to evolve, Application Programming Interfaces (APIs) are widely used to connect heterogeneous systems. The RESTful API approach enables schedule and attendance data to be presented in a standard HTTP- and JSON-based format, making it accessible to various clients, such as web and mobile applications, as well as third-party services (Dan et al., 2022)(Pangestu et al., 2022). Several previous studies have examined the implementation of web- and mobile-based academic information systems and attendance systems, including those utilizing the Laravel framework and QR code technology (Qr-code et al., 2021)(Mudaparsyah et al., 2023)(Ghany & Raharjo, 2025). However, most of them still focus on monolithic applications tied to a specific interface,

so their flexibility in integrating with other systems remains limited (Andema & Informasi, 2021)(Santoso, 2024).

Research This study proposes the development of a RESTful API for managing student schedules and attendance in a university setting, utilizing the Laravel framework as the backend. The API is designed to provide structured and consistent services for managing data on students, faculty, courses, schedules, and attendance records. Unlike previous studies, which generally integrated attendance logic directly into specific web or mobile applications (Hamdani et al., 2024)(Mudaparsyah et al., 2023), This study emphasizes a resource-oriented service design, a clear separation between the service layer and the user interface, and API functional testing using Postman-assisted test scenarios as the basis for functional and baseline performance evaluation (Ghany & Raharjo, 2025).

To keep the research focused, the scope of development is limited to core data CRUD operations and attendance tracking, without addressing integration with existing academic systems or comprehensive multi-role access management. Given these limitations, the objective of this research is to design a data model and implement a RESTful API for student schedule and attendance management, as well as to test the basic functionality and response times of the resulting service as a foundation for the development of a more integrated academic system in the future.

2. Research Methods

This study falls under the category of applied research, focusing on the development of a RESTful API to support the management of student schedules and attendance in a university setting (Pernando, 2021). The development method used follows a sequential process comprising four main stages: requirements analysis, system design, implementation, and testing, as shown in Figure 1. This phased approach aligns with various studies on RESTful API development in the domains of tracer studies, inventory management, and operational data collection, which generally begin with needs identification, architecture and data model design, API service implementation, and conclude with functional testing and basic performance evaluation (Anjarwani et al., 2022)(Saputra & Fery, 2024). In addition, system performance and responsiveness were also taken into account as part of the development process, as highlighted in the research on the optimization of web-based attendance applications (Sadikin et al., 2025).

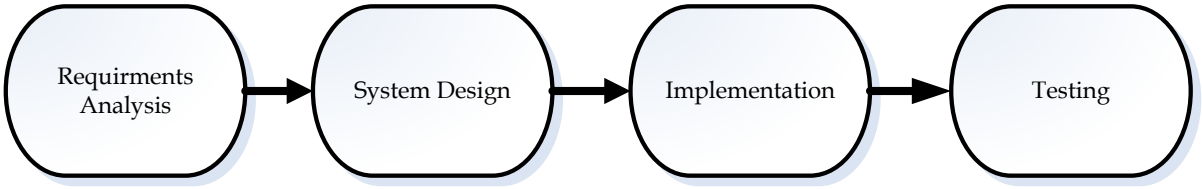


Figure 1. Development Method

2.1 Requirements Analysis

The requirements analysis phase focuses on identifying the system's functional and non-functional requirements. Functional requirements include the management of student, faculty, and course data, class schedules, and attendance tracking for each session. This approach aligns with the design patterns of attendance systems and other operational information systems that aim to replace manual or semi-manual processes (such as the use of paper attendance sheets or separate records) with centralized web- and mobile-based services (Lesmideyarti et al., 2023)(Dwi et al., 2024). Non-functional requirements include integration capabilities via RESTful APIs, consistent data formats (e.g., JSON-based), ease of access from various clients, and acceptable response times even as the number of requests increases. An emphasis on interoperability and cross-platform integration is also evident in the development of RESTful APIs for tracer studies and other operational management systems, where the APIs are

designed to serve as a bridge between databases and various client applications (Anjarwani et al., 2022)(Praptiwi et al., 2024). The results of this requirements analysis are then used to determine the API's primary resources and the operations that must be provided.

2.2 System Design

The system design phase involves creating a data model in the form of a relational database schema that represents core entities such as students, lecturers, courses, schedules, and attendance records along with their relationships, as well as designing a RESTful API service by defining CRUD endpoints and specialized attendance endpoints, including URL patterns, request parameters, JSON response formats, and HTTP status codes, as implemented in the design of academic information systems, attendance systems, and various RESTful APIs for tracer studies, inventory management, and vessel registration (Anjarwani et al., 2022)(Saputra & Fery, 2024).

2.4 Implementation

Implementation The system was implemented using Laravel as the primary backend, where each designed resource is implemented as a model, migration, and controller, enabling CRUD operations and attendance tracking to be accessed via consistent RESTful endpoints. This approach applies the principle of separation of concerns by separating business logic from routing and the user interface, utilizing the framework's built-in validation and database management features, as well as establishing a code structure that is easily integrated with both web and mobile applications, similar to various academic information systems, operational data collection systems, and employee/teacher attendance systems in previous research (Lesmideyarti et al., 2023)(Dwi et al., 2024).

2.2 Testing

The testing phase aims to verify that all RESTful API endpoints function according to specifications through a black-box testing approach, by testing API responses based on various input scenarios using Postman for key operations (creation, update, deletion, and retrieval of data for students, faculty, courses, schedules, and attendance), then evaluating the validity of the JSON response structure and content, HTTP status codes, baseline response times, and the consistency of endpoint behavior, in line with REST API testing practices for web-based operational applications that emphasize system functionality and responsiveness (Praptiwi et al., 2024)(Sadikin et al., 2025).

3. Results and Discussion

This chapter presents the results of applying the methods described in the previous section, covering the stages of requirements analysis, system design, implementation, and testing of the developed RESTful API. For each stage, the outcomes are detailed, ranging from the formulation of functional and non-functional requirements, the design of the data model and API endpoints, the implementation of services using the Laravel framework, to the test results obtained using Postman. In addition to presenting the results, this chapter also includes a discussion linking the implementation findings to the research objectives, so that it can be assessed to what extent the built system has met the needs for student schedule and attendance management.

3.1 Requirements Analysis

The requirements analysis phase produces a list of functional and non-functional requirements that serve as the basis for system design.

1. The formulated functional requirements include:
   a. The system is capable of managing student data, including adding, updating, deleting, and viewing student data.

b. The system is capable of managing faculty data, including adding, updating, deleting, and viewing faculty data.
c. The system is capable of managing course data, including course codes, course names, credit hours, and recommended semesters.
d. The system is capable of managing class schedules that link courses with instructors, as well as information on the day, time, and classroom.
e. The system can record student attendance for specific schedules, including attendance status (present, excused, sick, absent) and arrival time.
f. The system provides a feature to display attendance summaries by student and by class schedule.
g. All of the above functions can be accessed via a RESTful API, allowing them to be utilized by different client applications.

2. The formulated non-functional requirements include:
   a. Availability and reliability: the API service must run stably in the development environment and be easily deployable to the production environment.
   b. Performance: response times for basic operations (reading and adding data) should be kept low to ensure a smooth user experience for clients.
   c. Basic security: the API structure is designed to allow for the easy addition of authentication and authorization mechanisms during future development. Interoperability: data is transmitted in JSON format via the HTTP protocol so that it can be integrated with various platforms (both web and mobile).

3.2 Design System

The implementation of the RESTful API in this study uses the Laravel framework on the server side. The database used is MySQL to store student schedule and attendance data. Figure 2 shows the Entity Relationship Diagram (ERD) that illustrates the database structure and the relationships between the main entities, namely students, instructors, courses, schedules, and attendance. This ERD serves as a reference for creating migrations and Eloquent models in Laravel.

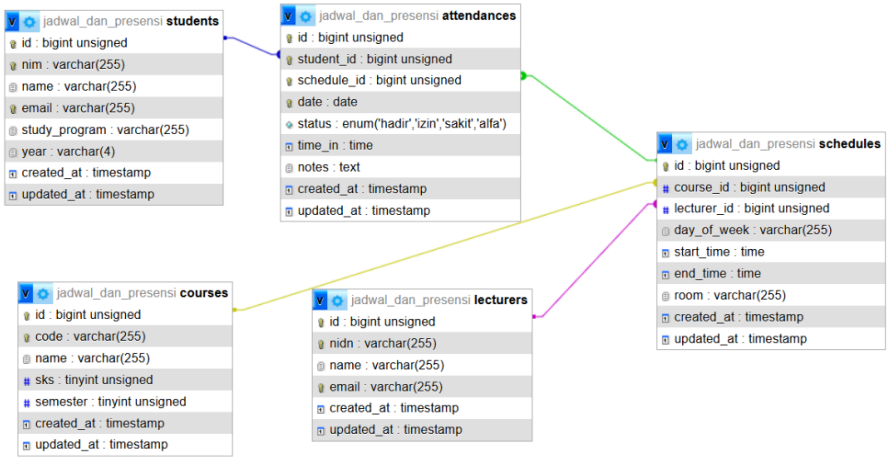


Figure 2. Database Design

Here is a brief explanation of each database table used:

1. students Table
   Used to store student data. The main attributes in this table include id as the primary key, nim as the student ID, name as the student's name, email as the email address, study_program as the academic program, and year as the graduation year. The created_at and updated_at attributes are used by Laravel to record the creation and update times of the data.
2. Lecturers Table
   Used to store data on lecturers involved in the teaching process. The attributes used include `id` as the primary key, `nidn` as the national lecturer ID, `name` as the lecturer's name, and

`email` as the email address. Like other tables, `created_at` and `updated_at` are used to record the time of data changes.

3. Courses Table
   Used to store course information. Attributes in this table include id, code as the course code, name as the course name, sks as the number of semester credit units, and semester as the recommended semester for offering the course. The created_at and updated_at attributes are used to track data changes.
4. The schedules table
   Used to store class schedule data. This table links courses and instructors via the course_id and lecturer_id attributes, which act as foreign keys to the courses and lecturers tables. Additionally, it includes the day_of_week attribute to store the day of the class, start_time and end_time to record the start and end times, and room to store classroom information.
5. Attendances Table
   Stores student attendance data for each class session. The student_id and schedule_id attributes act as foreign keys linking attendance to students and schedules. The date attribute is used to record the class date, status stores the attendance status (present, excused, sick, or absent), time_in records the arrival time, and notes stores additional remarks if needed. This database structure is then mapped into a RESTful API service using Eloquent models and controllers in Laravel. Each main table is represented as a resource accessible via endpoints prefixed with /api, such as /api/students for student data, /api/lecturers for instructor data, /api/courses for course data, /api/schedules for schedule data, and /api/attendances for attendance data. Communication between the client and server utilizes standard HTTP methods (GET, POST, PUT, and DELETE) with a JSON-based data exchange format, ensuring that the built service is lightweight, consistent, and easily integrable with various client applications in the future (Izza et al., 2022)(Polgan et al., 2023)(Irmaliantie et al., 2024).

3.3 Implementation

The system was implemented using the Laravel framework as the backend and MySQL as the database. The implementation process began with the creation of migrations and models for each main entity, followed by the creation of RESTful controllers and the definition of routes in the routes/api.php file. This entire process resulted in an API service accessible via the address http://localhost/api/... in the development environment.

1. Implementation of Models and Migrations
   Each entity in the ERD is implemented as a table via migration files in the database/migrations directory. For example, the students table is implemented with the attributes nim, name, email, study_program, and year defined as string-type columns, and id as a primary key of type big integer unsigned. Laravel automatically adds the `created_at` and `updated_at` columns to record the creation and update times of the data. After the migrations are created, Eloquent models for `Student`, `Lecturer`, `Course`, `Schedule`, and `Attendance` are created in the `app/Models` directory. Each model contains a $fillable property to define attributes that can be bulk-populated, as well as declarations of relationships between entities, such as the hasMany relationship between Student and Attendance, and the belongsTo relationship between Attendance and Schedule. Defining these relationships simplifies the process of retrieving interconnected data without having to explicitly write SQL queries.
2. Controller and Routing Implementation
   Business logic for each *resource* is implemented in the *controller* located in the App\Http\Controllers\Api namespace. For example, the StudentController shown in Figure 2 contains the index, store, show, update, and destroy methods, which handle operations such as retrieving a list of

students, adding new data, displaying details, updating, and deleting data. Each method returns a response in a consistent JSON format, for example, containing a *flag* for success, a *status* message, and a data object containing the operation results. In the AttendanceController, two additional methods—byStudent and bySchedule—have been added to generate attendance summaries based on students or class schedules.

app > Http > Controllers > Api > StudentController.php > ...

```php
class StudentController extends Controller
{
    public function index()
    {
        // Bisa ditambah pagination kalau mau
        $students = Student::all();

        return response()->json([
            'success' => true,
            'data' => $students,
        ]);
    }

    public function store(Request $request)
    {
        $validated = $request->validate([
            'nim'           => 'required|string|unique:students,nim',
            'name'          => 'required|string',
            'email'         => 'required|email|unique:students,email',
            'study_program' => 'nullable|string',
            'year'          => 'nullable|string|max:4',
        ]);
```

Figure 3. Code Snippet from StudentController

All RESTful services are registered via the routes/api.php file shown in Figure 3. In this file, the Route::apiResource class is used to define standard CRUD routes for the students, lecturers, courses, schedules, and attendances resources.

Additionally, specific routes are defined for attendance summaries, namely GET /api/students/{student}/attendances and GET /api/schedules/{schedule}/attendances. With this approach, all API endpoints are centralized in a single routes file and automatically use Laravel's built-in API middleware.

```php

// Resource API
Route::apiResource('students', StudentController::class);
Route::apiResource('lecturers', LecturerController::class);
Route::apiResource('courses', CourseController::class);
Route::apiResource('schedules', ScheduleController::class);
Route::apiResource('attendances', AttendanceController::class);

// Endpoint tambahan untuk rekap presensi
Route::get('students/{student}/attendances', [AttendanceController::class, 'byStudent']);
Route::get('schedules/{schedule}/attendances', [AttendanceController::class, 'bySchedule']);

```

Figure 4. Code Snippet from StudentController

3.4 Testing

System testing was conducted to ensure that all RESTful API endpoints functioned in accordance with the defined requirements. Testing was performed in a local development environment using `php artisan serve` as the web server and Postman as the tool for sending HTTP requests. The testing focused on CRUD operations for each resource, as well as the recording and summarization of student attendance.

Test scenarios were designed to follow the system's usage flow. The first phase involved testing the students, lecturers, and courses endpoints to ensure proper management of master data. The next phase tested the schedules endpoint to ensure class schedules could be generated from combinations of

courses and instructors. The final phase tested the attendances endpoint, as well as the attendance summary endpoints for each student and each schedule.

In each scenario, POST requests are used to add data, GET to read data, PUT to update data, and DELETE to delete data. In addition to examining the response body, the researchers also observed the HTTP status codes returned and the response times recorded in Postman.

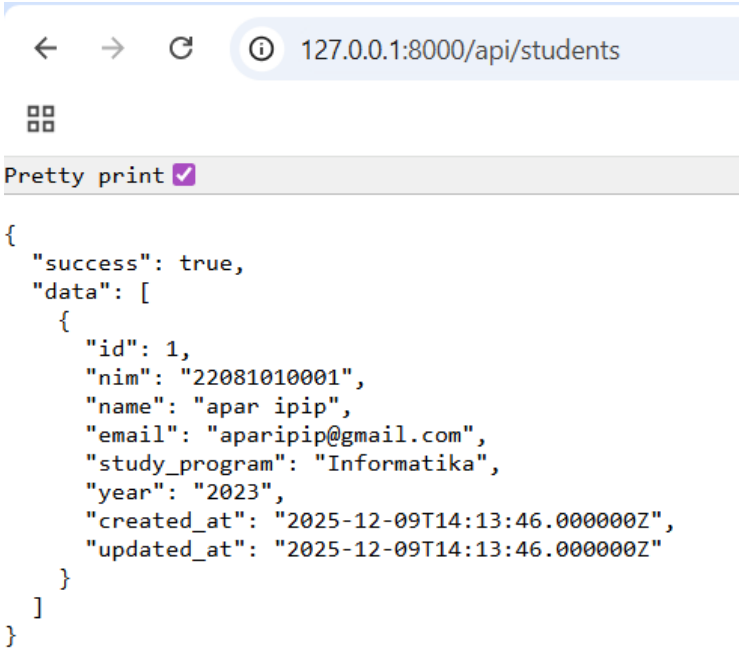
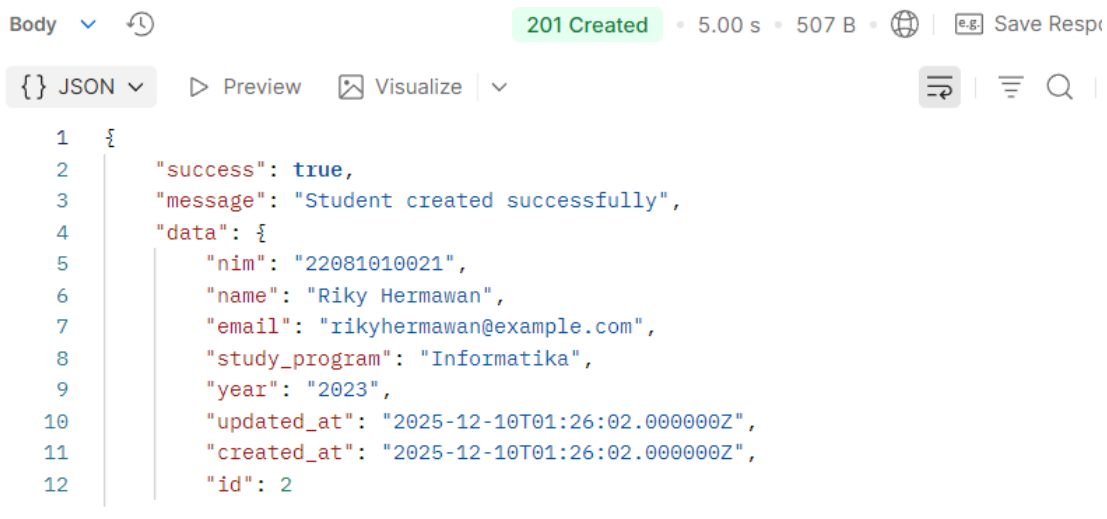


Figure 5. Testing the GET endpoint /api/students.

In each scenario, a POST request is used to add data, a GET request to read data, a PUT request to update data, and a DELETE request to delete data. In addition to examining the response body, the researcher also observed the HTTP status codes returned and the response times recorded in Postman.

CRUD functionality testing was performed on the students, lecturers, and courses resources using Postman. For each resource, the testing sequence began with data insertion (POST) (Figure 6), followed by retrieving the list and details of data (GET) (Figure 6), updating data (PUT) (Figure 7), and finally deleting data (DELETE) (Figure 8). CRUD function testing on the students resource begins with a GET request to /api/students to ensure the service can return a list of students in JSON format. The response received is a JSON object containing data elements and a 200 OK status code (Figure 5).



Gambar 6. Testing the POST endpoint /api/students in Postma.

As shown in Figure 6, the POST request to /api/students sent via Postman returns a response with a status code of 201 Created, indicating that the student data was successfully added. The returned response is a JSON object containing a `success` attribute with a value of `true`, the message "Student created successfully", and a data object containing details of the newly saved student, such as `nim`, `name`, `email`, `study_program`, and `year`, along with the `id`, `created_at`, and `updated_at` values automatically populated by the system. This indicates that the POST endpoint /api/students is functioning as intended to add new student data to the database

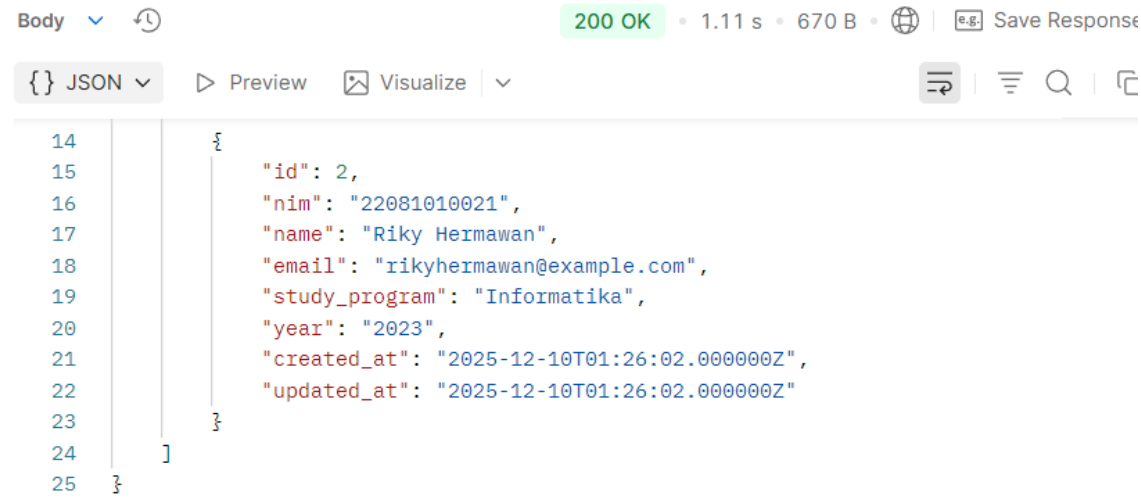


Figure 7. Testing the GET /api/students endpoint in Postman.

Figure 7 shows the results of testing the GET /api/students endpoint using Postman. The response was returned with a 200 OK status code, indicating that the service successfully retrieved a list of students from the database in accordance with the functional requirements for the data read operation.

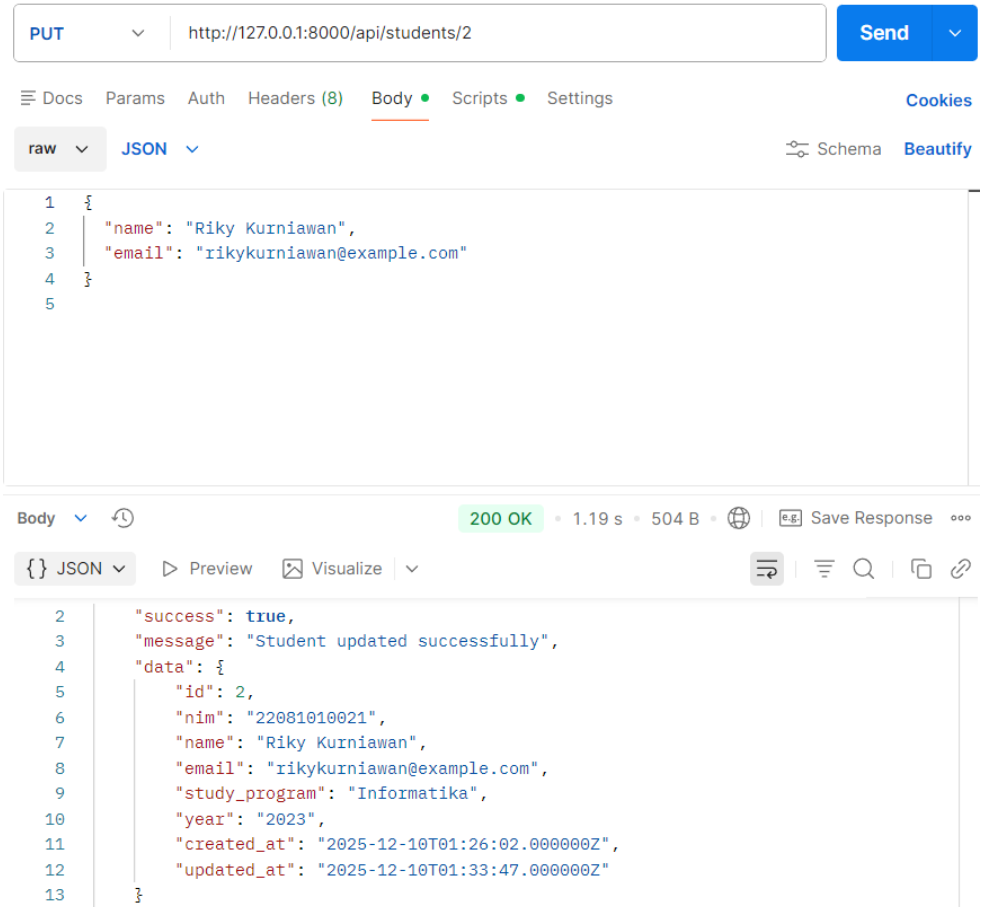


Figure 8. Testing the PUT endpoint /api/students in Postman

Figure 8 shows the results of testing the PUT endpoint /api/students/2 to update the data for the student with ID 2. The request was sent with changes to the name and email attributes, and the system responded with a 200 OK status code and the message "Student updated successfully". The data object shows the student's values after the update and the change in the `updated_at` attribute, indicating that the data update operation via the PUT endpoint `/api/students/{id}` was successful.

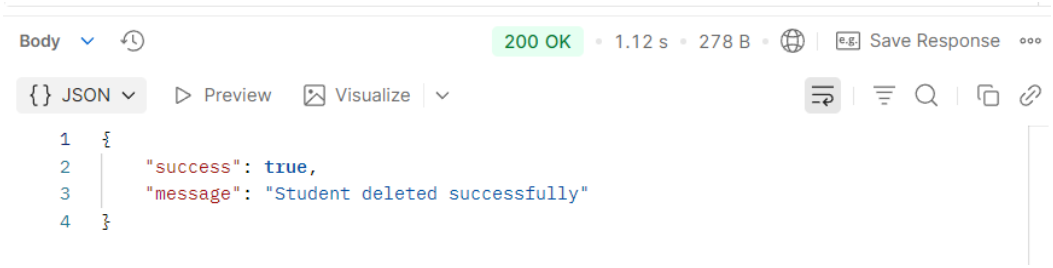


Figure 9. Testing the DELETE endpoint /api/students in Postman

Figure 9 shows the results of testing the DELETE endpoint /api/students/2. The request to delete the student data with ID 2 returned a 200 OK response with the message "Student deleted successfully." This confirms that the DELETE endpoint /api/students/{id} is functioning properly to delete student data from the system.

4. Conclusion

Based on the series of design, implementation, and testing activities that have been carried out, it can be concluded that the development of a RESTful API service for student schedule and attendance management using the Laravel framework and a MySQL database has been successfully implemented in accordance with requirements. This success aligns with various implementations of academic and attendance information systems based on Laravel and RESTful architecture, which demonstrate the effectiveness of similar approaches in managing academic and attendance data in an integrated manner within educational environments (Luh et al., 2023)(Studi & Informasi, 2025)(Uddin et al., 2025). The designed data model is capable of representing student, faculty, course, schedule, and attendance entities in a structured manner, so that relationships among academic data can be managed clearly and consistently, while the implementation of services via RESTful endpoints enables CRUD operations on master data as well as attendance recording and summarization to be performed through a simple workflow that remains aligned with academic business processes, as is also applied in employee attendance systems and REST API-based inventory management applications (Rizkyansah & Dzikri, 2025)(Yudanto et al., 2025). Testing results using Postman show that each endpoint returns a response in JSON format with the appropriate HTTP status code; thus, the built API can be considered functionally stable in the local development environment and aligns with established practices for functional endpoint testing in REST API backend development, as documented in various previous studies (Rumonang et al., 2025)(Susilowati & Voutama, 2025). Separating business logic into RESTful API services and using the JSON data exchange format provides flexibility for future web and mobile interface development, as well as opening up opportunities for integration with other academic information systems already in use at universities, as also emphasized in research related to the development of SIAKAD, web-based attendance systems, application architecture optimization, and interface design that adapts to user needs (Farchani et al., 2025)(Rahayu et al., 2025)(Dhaifullasyah et al., 2024)..

.

References

Andema, H., & Informasi, S. (2021). Rancangan sistem informasi akademik sekolah berbasis laravel. *Jurnal Sistem Informasi Dan Manajemen*, *9*(1), 99–108. https://doi.org/https://doi.org/10.47024/js.v9i1.297

Anjarwani, S. E., Irmawati, B., Agitha, N., Afwani, R., Studi, P., Informatika, T., Teknik, F., Mataram, U., Studi, P., Informatika, T., Teknik, F., Mataram, U., Studi, P., Informatika, T., Teknik, F., Mataram, U., Studi, P., Informatika, T., Teknik, F., … Mataram, U. (2022). IMPLEMENTASI RESTFUL API PADA SISTEM

INFORMASI TRACER STUDY UNIVERSITAS MATARAM BERBASIS MOBILE. *Prosiding SAINTEK LPPM Universitas Mataram*, *4*(November 2021), 23–24.

Dan, A., Restful, I., Guna, A. P. I., Informasi, S., Pada, A., & Tinggi, P. (2022). ANALISIS DAN IMPLEMENTASI RESTFUL API GUNA PENGEMBANGAN SISTEM INFORMASI AKADEMIK PADA PERGURUAN TINGGI. *Jurnal Informatika Terpadu*, *8*(1), 47–61. https://doi.org/https://journal.nurulfikri.ac.id/index.php/JIT

Dhaifullasyah, A. T., Irfan, D., Marta, R., & Farell, G. (2024). Rancang Bangun Sistem Absensi Menggunakan Framework Laravel Yang Terintegrasi Berbasis Rfid Dan Face Recognition Untuk Smkn 1 Tilatang Kamang. *JURNAL TEKNIK KOMPUTER DAN INFORMATIK*, *4*(2), 46–51. https://doi.org/https://doi.org/10.24036/jteki.v4i2%7CAugust.67

Dwi, P., Pangestu, A., Permatasari, H., & Widyaningsih, P. (2024). Sistem Informasi Presensi Karyawan Menggunakan Qr Code Berbasis Web Pada PT Berkat Bagi Sesama Kota Surakarta. *JURNAL TEKNIK INFORMATIKA*, *4*(3), 568–579. https://doi.org/10.58794/jekin.v4i3.845

Farchani, S. B., Kusuma, B. A., & Hermanto, N. (2025). Implementasi rest api dalam pengembangan backend inventory peminjaman. *Jurnal Ilmiah Penelitian Dan Pembelajaran Informatika*, *10*(2), 1404–1413. https://doi.org/https://doi.org/10.29100/jipi.v10i2.6249

Ghany, S. B., & Raharjo, G. (2025). SISTEM INFORMASI PRESENSI SISWA MENGGUNAKAN QR CODE BERBASIS WEB (STUDI KASUS : MTS . ASSALAFIYAH KOTA TEGAL). *Jurnal Mahasiswa Teknik Informatika*, *9*(5), 7874–7880. https://doi.org/https://doi.org/10.36040/jati.v9i5.14931

Hamdani, D., Purno, A., Wibowo, W., & Heryono, H. (2024). Perancangan Sistem Presensi Online dengan QR Code Menggunakan Metode Prototyping. *Jurnal Teknologi Fan Informasi*, *14*, 62–73. https://doi.org/10.34010/jati.v14i1

Irmaliantie, D., Hamid, A., Naury, C., Informatika, M., Harapan, P., & Surakarta, B. (2024). Sistem Informasi Presensi Berbasis Android Menggunakan QR Code Pada Kantor Desa Pandawan. *Journal of Information Technology Dn Conputing*, *4*(1), 210–222. https://doi.org/https://doi.org/10.52187/img.v4i1.131

Izza, M., Latansya, A., Arwani, I., & Brata, D. W. (2022). Pengembangan Sistem Informasi Pencatatan Nilai dan Presensi berbasis Website pada Rumah Sakit Umum Daerah Kanjuruhan Kabupaten Malang. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *6*(7), 3471–3480. https://doi.org/https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/11374

Lesmideyarti, D., Yanti, N., Elektro, J. R., & Balikpapan, P. N. (2023). PERANCANGAN DAN IMPLEMENTASI SISTEM INFORMASI PRESENSI GURU DAN TENAGA KEPENDIDIKAN MENGGUNAKAN FRAMEWORK CODEIGNITER. *JURNAL SAINS TERAPAN*, *9*(2), 1–7. https://doi.org/https://doi.org/10.32487/jst.v9i2.1879

Luh, N., Pivin, G., Adi, P., Permana, G., Aditya, P., Prayoga, A., & Kadek, N. (2023). Implementasi Framework Laravel Pada Sistem Informasi Akademik SMA Negeri 1 Kediri Berbasis Web. *Jurnal Nasional Komputasi Dan Teknologi Informasi*, *6*(3), 260–267. https://doi.org/https://doi.org/10.32672/jnkti.v6i3

Mudaparsyah, M., Rusdini, D., Taryanto, A., Studi, P., Informasi, S., Studi, P., Informatika, M., & Ganesha, P. P. (2023). IMPLEMENTASI SISTEM INFORMASI PRESENSI MAHASISWA MENGGUNAKAN FRAMEWORK LARAVEL DI POLITEKNIK PIKSI GANESHA. *Jurnal of Information Technology Student*, *2*, 89–97.

Pangestu, H. A., Kurniadi, D., & Septiana, Y. (2022). *Aplikasi Pengelolaan Data Pegawai Berbasis REST API untuk Transfer Data Real Time dengan Framework Codeigniter*. *19*, 313–322. https://doi.org/https://doi.org/10.33364/algoritma/v.19-1.1090

Pernando, J. (2021). Sistem Absensi Online Berdasarkan GPS Menggunakan Framework Laravel. *JURNAL TEKNIK INFORMATIKA*, *1*(1), 40–49. https://doi.org/https://doi.org/10.58794/jekin.v1i1.23

Polgan, J. M., Guntara, R. G., Azkarin, V., & Indonesia, U. P. (2023). Implementasi dan Pengujian REST API Sistem Reservasi Ruang Rapat dengan Metode Black Box Testing (Studi Kasus : PT Lizzie Parra Kreasi). *Jurnal Minfo Polgan*, *12*, 1229–1238. https://doi.org/https://doi.org/10.33395/jmp.v12i1.12691 e-ISSN

Praptiwi, R. A., Priandika, A. T., & Alita, D. (2024). Implementasi REST API pada Manajemen Stok Barang Berbasis Aplikasi Web (Studi Kasus : PT Jon Kuliner Indonesia). *Jurnal Teknik Komputer*, *3*(1), 19–24. https://doi.org/10.14710/jtk.v3i1.46234

Qr-code, M. T., Ardiansyah, I., Saputro, W. T., & Widatama, K. (2021). Pengembangan Sistem Informasi Presensi Mahasiswa Memanfaatkan Teknologi Qr-Code. *Jurnal INTEK*, *4*, 91–100. https://doi.org/https://doi.org/10.37729/intek.v4i2.1699

Rahayu, S., Sutedi, A., & Mutiara, C. (2025). Implementasi Rest Api Untuk Pencatatan Akta Kelahiran dan Kematian Pada Platform Web. *Jurnal Algoritma*, *22*, 336–346. https://doi.org/10.33364/algoritma/v.22-1.1704

Rizkyansah, S., & Dzikri, M. T. P. A. (2025). ANALISIS DAN PERANCANGAN SISTEM PRESENSI KARYAWAN PT BERKAH UNTUK SEMESTA. *Jurnal of Information System and Business Technology*, *1*(1), 1–7. https://doi.org/https://journal.jci.co.id/jisbt/article/view/3

Rumonang, D., Anatasya, A. E. F., & Wahyuni, E. D. (2025). Modul Orang Tua pada Web Sistem Informasi Universitas Handayani Makassar M enggunakan Laravel Parent Module in the Handayani University Makassar Information System Website using Laravel Framework. *Jurnal Sistem Informasi*, *14*, 1397–1405. https://doi.org/https://doi.org/10.32520/stmsi.v14i3.5191

Sadikin, A., Rahim, A., Siswanto, A., Wardani, M., & Adiputra, R. F. (2025). Optimasi Kecepatan dan Responsivitas Aplikasi Presensi Perkuliahan Berbasis Web melalui Arsitektur Single Page Application. *Jurnal Ilmiah MEDIA SISFO*, *19*(2), 262–269. https://doi.org/https://doi.org/10.33998/mediasisfo.2025.19.2.2539

Santoso, M. H. (2024). PENGEMBANGAN SISTEM INFORMASI AKADEMIK BERBASIS WEB MENGGUNAKAN FRAMEWORK LARAVEL. *Jurnal Informatika Terpadu*, *10*, 1–13. https://doi.org/https://doi.org/10.54914/jit.v10i2.1436

Saputra, L. S., & Fery, H. H. (2024). IMPLEMENTASI REST API PADA SISTEM PENDATAAN KAPAL MENGGUNAKAN LARAVEL 9 DAN NUXTJS. *Jurnal Conputer and Technologi*, *2*(1), 10–21. https://doi.org/IMPLEMENTASI REST API PADA SISTEM PENDATAAN KAPAL MENGGUNAKAN LARAVEL 9 DAN NUXTJS

Studi, P., & Informasi, S. (2025). PENGEMBANGAN SISTEM INFORMASI AKADEMIK PADA SMP NEGERI 3 TANAH PUTIH MENGGUNAKAN FRAMEWORK LARAVEL. *Jurnal Sistem Informasi Bisnis*, *6*(1), 33–42. https://doi.org/https://doi.org/10.55122/junsibi.v6i1.1534

Surakarta, S. A. U. B., Kardha, D., Sumboro, B., & Setyawan, A. E. (2018). Sistem Informasi Presensi Mahasiswa pada Sekolah Tinggi Manajemen Informatika Dan Komputer " Adi Unggul Bhirawa ." *JURNAL ILMIAH STMIK AUB*, *24*(1), 41–53. https://doi.org/10.36309/ goi.v24i1.86

Susilowati, S. C., & Voutama, A. (2025). Desain UI / UX Adaptif Website Absensi Karyawan : Komparasi Desktop dan Mobile Dengan ACD. *Jurnal Ilmu Komputer Dan Bisnis*, *XVI*(2), 11–22. https://doi.org/https://doi.org/10.47927/jikb.v16i2a.1075

Syifa, N., Yusdhistira, Y., & Nabyla, F. (2025). Sistem Informasi Presensi Berbasis QR Code menggunakan Framework Laravel pada SMK Ma ' arif Nu Tonjong. *Jurnal Sistem Informasi Dan Teknologi Peradaban*, *6*(1), 28–36. https://doi.org/https://doi.org/10.58436/jsitp.v6i1.2237

Uddin, M. B., Susanto, A., & Hamdani, A. (2025). Implementasi rfid pada sistem informasi presensi siswa di mas al-huda kangayan dengan notifikasi whatsapp gateway. *Jurnal Teknologi Dan Sistem Informasi Univrab*, *10*(2), 717–725. https://doi.org/https://doi.org/10.36341/rabit.v10i2.6361

Yudanto, B. A., Laudri, S. P., Felina, I., Rizqi, M., Abinovan, A., Fadhil, M., Fathir, R., Putra, R., Ulfa, D. F., Salzi, W. F., Nuzulah, R., Studi, P., Informatika, T., Gedong, K., Rebo, P., & Timur, J. (2025). IMPLEMENTASI REST API MENGGUNAKAN BAHASA GO UNTUK OPTIMALISASI MANAJEMEN MENU DAN PEMESANAN. *Jurnal Risat Dan Aplikasi Mahasiswa Informatika*, *06*(02), 384–393. https://doi.org/https://doi.org/10.30998/jrami.v6i02.13367