Vol. 6 No. 1, 2025, pp. 71-80
Jurnal Pepadun

DOI: 10.23960/pepadun.v6i1.257

Development of RESTful API in Lecturer Remuneration System at The University of Lampung Using JWT Authentication

[*1]M. Iqbal Parabi, [2]Rizky Prabowo, [3]Igit Sabda Ilman, [4]Bayu Wicaksono, [5]Shafa Auliya and [6]Fathimah Abiyyi Khairunnisa

[1,2,3,4,5,6] Department of Computer Science, University of Lampung, Jalan Prof. Dr. Sumantri Brojonegoro Nomor 1, Bandar Lampung, Indonesia
e-mail: [*1]**iqbal.parabi@fmipa.unila.ac.id**, [2]rizky.prabowo@fmipa.unila.ac.id, [3]igit.sabda@fmipa.unila.ac.id, [4]2427051008@students.unila.ac.id, [5]shafa.auliya21@students.unila.ac.id, [6]fathimah.abiyyi21@students.unila.ac.id

Abstract *— In today's dynamic digital era, Universitas Lampung (Unila) faces challenges in managing increasingly complex information systems. With the rapid and diverse adoption of technology, system integration is essential to ensure seamless interconnectivity and operational efficiency. One of the most complex systems at Unila is the Lecturer Remuneration System, which evaluates lecturer performance in the areas of education, teaching, research, community service, and administrative support. This system generates valid data, assessed by appointed evaluators, in the form of lecturer work points. However, the utilization of data and information within the Lecturer Remuneration System remains limited to performance assessment purposes. In reality, validated research and service history are crucial for the accreditation process of academic programs. This highlights the need for system integration to optimize data and information utilization. This study aims to develop a system integration module using a RESTful API architecture, enabling the Lecturer Remuneration System to interface with other systems at Unila. To ensure data security, authentication based on JWT is implemented. JWT provides a compact and self-contained way to securely transmit information between parties, ensuring the integrity and confidentiality of data exchanges within the system. API testing results show that all tested API endpoints responded successfully with HTTP 200 status codes. Response times ranged from 349 ms to 1534 ms, and response sizes varied from 0.648 KB to 5.27 KB, indicating reliable performance and acceptable efficiency.*

Keywords: *API; Data; Information; Integration; Restful.*

1. INTRODUCTION

The rapid development of technology today provides convenience in various aspects of work, including in the field of education [1]. Educational institutions are required to adapt to these changes in order to improve service quality and achieve optimal educational outcomes. One of the efforts that can be made is to develop an information system that is able to manage activities quickly, effectively, efficiently and accurately [2]. The presence of an adequate information system allows educational institutions to achieve goals more optimally, especially in managing the performance and quality of educational services.

University of Lampung (Unila) as a higher education institution in Indonesia also faces great challenges in managing increasingly complex information systems. With the rapid and diverse adoption of technology, integration between systems is required to ensure each system is interconnected and works efficiently. One of the systems at Unila that has high complexity is the Lecturer Remuneration System, which is used to measure lecturer performance based on lecturer workload in the fields of education, teaching, research, service, and support [3]. The lecturer workload that must be met each semester ranges from a minimum of 12 credits and a maximum of 16 credits [4].

This system produces valid data and information that has been assessed by the assessors, with the output in the form of lecturer performance points. The points are then converted into the amount of rupiah that the lecturer will get in each semester as a remuneration allowance. However, after the conversion into the amount

DOI: 10.23960/pepadun.v6i1.257

of rupiah, lecturer performance data is only stored on the server and cannot be used by other systems that need it, such as systems that monitor performance in the fields of research and service.

The utilization of data and information in the Lecturer Remuneration System is currently not optimal, because it only functions as a measuring tool for lecturer performance to determine the amount of remuneration allowance. In practice, the performance history of lecturers in the field of research and service is needed by study programs or related work units, especially when carrying out the accreditation process. This limitation causes study programs that are conducting accreditation to experience difficulties in tracking lecturer research and service data effectively.

An integration system is a process that involves combining different subsystems or components into one large system. It ensures that each integrated subsystem functions as needed. Currently, RESTful API has become a widely used standard for creating APIs that can be accessed by other systems [5]. Its simplicity, scalability, and compatibility with various platforms make it a preferred choice for enabling secure and efficient data [6][7]. In the context of the Lecturer Remuneration System at the University of Lampung, RESTful APIs are essential to facilitate integration with other institutional systems such as Quality Assurance System Unila, allowing seamless data sharing across administrative, academic, and accreditation workflows. RESTful API has the main advantage of high flexibility and capability, making it easy to implement with various types of platforms [8]. By utilizing RESTful APIs, lecturer performance data can be accessed by other systems efficiently.

In addition to the need for integration, data security is also an important factor in the development of this system. Lecturer performance data, which includes sensitive information, needs to be protected from unauthorized access. For this reason, the implementation of JSON Web Token (JWT) is very appropriate as one of the steps to increase the level of security. JWT functions as a token that contains information used for the authentication process and information exchange [9]. With that, only parties with authorized tokens can access the data, ensuring the protection of very important information.

Therefore, it is necessary to integrate the Lecturer Remuneration System with other systems to maximize the utilization of available data and information. This integration will not only facilitate access to lecturer performance data in the field of research and service, but also allow optimal utilization of data for various other needs. With secured integration, the accreditation process can run more efficiently.

2. RESEARCH METHODOLOGY

2.1 Identify Needs

The identification of API development requirements involves defining system goals, identifying stakeholders, gathering functional and non-functional needs, analyzing data sources, and drafting API specifications. This process ensures that the API meets performance, security, and usability expectations before implementation. The data used in this research were obtained from existing institutional databases and include structured information such as lecturer profiles, teaching, research, community service, and administrative support. The data types involved consist of strings, integers, dates, and arrays, all of which are stored in a relational database and exchanged via JSON format through standard HTTP methods.

2.2 Literature Study

At this stage, a literature study is carried out to collect information relevant to the research to obtain a theoretical basis for the problem to be studied, such as scientific articles, previous research journals, reference books, and other supporting documents. In addition, the literature study aims to make the research more directed, focused, and avoid similarities with previous research, so that it can make a significant contribution to the field being studied.

Vol. 6 No. 1, 2025, pp. 71-80 **Jurnal Pepadun**

DOI: 10.23960/pepadun.v6i1.257

2.3 API Design

The API design stage aims to determine how the API will optimally expose data and functionality to users. At this stage, developers draft API specifications in a standards-compliant format, such as OpenAPI. These standards provide a formal framework for designing APIs, documenting functionality, and testing API.

2.4 API Implementation

At this stage, it is done by implementing the API according to predetermined specifications. The code is developed using a modular approach to ensure flexibility and ease of maintenance. A JWT-based authentication system is integrated to enhance the security of API access.

2.5 API Testing

The testing phase aims to ensure that the API functions according to the design specifications and user requirements. Testing is performed iteratively, starting at the development stage through to the deployment stage, using both manual and automated approaches. These tests are applied to validate that the API remains consistent with the specifications set at the design stage. In addition, Functional Testing is performed to ensure that each API endpoint operates according to the specific needs of the user.

To ensure the reliability of the API in the face of real workloads, Performance Testing is performed to measure the response time and stability of the API under various conditions. With thorough and structured testing, potential issues can be identified early, reducing the risk of failure in the production environment. This ensures that the APIs released are high quality, secure and reliable for users.

3. RESULTS AND DISCUSSION

3.1 Identify Needs

Requirements identification is the stage of collecting and defining functional and non-functional requirements on the development of the Unila lecturer remuneration system API. The identification process is carried out through analysis of the business processes contained in the Unila lecturer remuneration system using the use case diagram which can be seen in Figure 1.

Figure 1 is a use case diagram of the Unila lecturer remuneration system. This use case involves four actors, namely admin, dean, head of study program, and lecturer, with each having different roles and access rights in the system. The admin is responsible for managing accounts, including adding, changing, and deleting user accounts as needed. Furthermore, the dean has access to view overall lecturer performance data, covering all study programs in the faculty. On the other hand, the head of study program can only access the performance data of lecturers who are within the scope of the study program they lead. Lecturers can view their own performance data, which allows lecturers to monitor their progress and achievements in the fields of education, research, service, and support.

3.2 Literature Study

The literature study on RESTful API with JWT authentication is an efficient approach to building secure distributed systems. RESTful API provides a framework that allows communication between systems via HTTP protocol with GET, POST, PUT, or DELETE methods [10]. Meanwhile, JWT performs an authentication and information exchange system that ensures each request has valid token to access data [11].

Previous research, such as that conducted by Nashikhuddin, et al., discussed the application of RESTful API with JWT for user authentication and authorization in the Thrift Shop E-commerce application. This research aims to improve system security and performance by ensuring sensitive data access is only granted to properly authenticated users [12].

Vol. 6 No. 1, 2025, pp. 71-80 **Jurnal Pepadun**

DOI: 10.23960/pepadun.v6i1.257

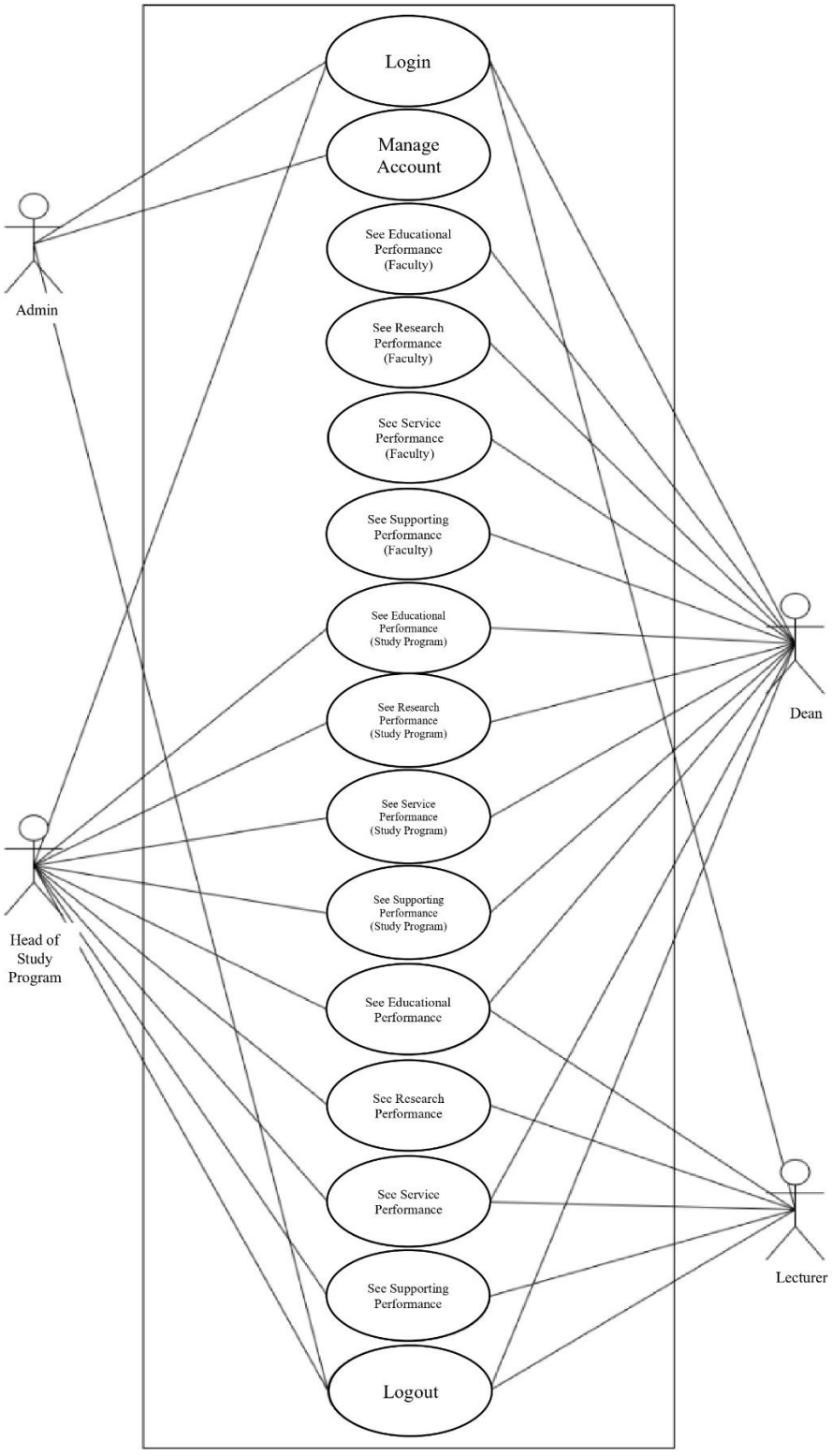

Figure 1. Use case diagram.

Furthermore, Pattinama, et al., developed a RESTful API with JWT authentication in a property application to secure data access. In this research, the API verifies the token provided, grants access according to user permissions, and ensures data security. This implementation not only improves communication between platforms, but also strengthens coordination in the property industry, and improves data security in Android applications [13].

Vol. 6 No. 1, 2025, pp. 71-80 **Jurnal Pepadun**

DOI: 10.23960/pepadun.v6i1.257

Gustiegan and Painem, discussed the development of a RESTful API service equipped with JWT authentication and AES-256 encryption to improve the security and efficiency of data exchange. Upon successful login, the system issues a JWT token that is used by the client to securely access the service. In addition to authentication, AES-256 encryption is applied to keep important data confidential during the information exchange process. This approach is successfully applied to the mobile-based laboratory loan system at Budi Luhur University, and supports interoperability between platforms such as mobile applications and backend servers [14].

With that, the implementation of RESTful API with JWT authentication is proven to be effective in improving the security, scalability, and interoperability of distributed systems. Various studies have shown that the integration of JWT and API not only ensures strict access control, but also facilitates communication between different platforms. With wider adoption, this technology is expected to support the development of more secure and efficient applications. Therefore, it has been proven that using RESTful APIs with JWT authentication can improve the security, scalability, and interoperability of distributed systems. According to various studies, combining JWT with APIs guarantees strict access control while facilitating communication between platforms. It is predicted that as this technology becomes more widely used, it will facilitate the creation of more secure and more effective applications.

3.3 API Design

The API design stage is used to define methods, endpoints, descriptions, requests, and parameters used before implementing the API. The API design in this study can be seen in Table 1 below.

Table 1. API design for the University of Lampung lecturer remuneration system

No	Method	Endpoint	Description	Request	Parameters
1	POST	/api/login	Login and generate JWT token.	JSON Body: { "email": "user@unila.ac.id", "password": "pass" }	-
2	GET	/api/pendidikan	Get performance data in the education sector.	Header: Authorization: Bearer {token}	fakultas, prodi, id_dosen, tahun, limit, offset, sort, order, search
3	GET	/api/pelaksanaan_ pendidikan	Obtaining performance data in the field of education implementation.	Header: Authorization: Bearer {token}	fakultas, prodi, id_dosen, tahun, limit, offset, sort, order, search
4	GET	/api/penelitian	Obtaining research field performance data.	Header: Authorization: Bearer {token}	fakultas, prodi, id_dosen, tahun, limit, offset, sort, order, search
5	GET	/api/pengabdian	Get performance data in the field of community service.	Header: Authorization: Bearer {token}	fakultas, prodi, id_dosen, tahun, limit, offset, sort, order, search
6	GET	/api/penunjang	Get supporting field performance data.	Header: Authorization: Bearer {token}	fakultas, prodi, id_dosen, tahun, limit, offset, sort, order, search
7	GET	/api/kinerja	Get lecturer performance data.	Header: Authorization: Bearer {token}	kode, fakultas, prodi, id_dosen, tahun, limit, offset, sort, order, search

Vol. 6 No. 1, 2025, pp. 71-80
Jurnal Pepadun

DOI: 10.23960/pepadun.v6i1.257

No	Method	Endpoint	Description	Request	Parameters
8	GET	/api/detil_kinerja	Get detailed data on lecturer performance.	Header: Authorization: Bearer {token}	id

3.4 API Implementation

API implementation is done by defining API specifications that have been designed at the API design stage. This research uses OpenAPI to define and document the API interface which can be seen in Figure 2. After defining the API specification using OpenAPI, API coding is done using the PHP programming language with the CodeIgniter 4.5.5 framework. The main function code snippet used to generate JWT tokens and view lecturer performance can be seen in program Code 1 and Code 2.

```yaml
openapi: 3.0.3
info:
  title: Kinerja Dosen API
  description: API untuk mendapatkan data kinerja dosen berdasarkan BKD
  version: 1.0.0
servers:
  - url: http://localhost:8080

paths:
  /api/login:
    post:
      summary: Login dan menghasilkan token JWT
      requestBody:
        required: true
        content:
          application/json:
            schema:
              type: object
              properties:
                email:
                  type: string
                  example: "user@fmipa.unila.ac.id"
                password:
                  type: string
                  example: "pass"
      responses:
        '200':
          description: Login berhasil
          content:
            application/json:
              schema:
                type: object
                properties:
                  message:
                    type: string
                    example: "Login successful"
                  token:
                    type: string
                    example: "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."
        '401':
          description: Unauthorized
```

Figure 2. API specification using OpenAPI.

Vol. 6 No. 1, 2025, pp. 71-80 **Jurnal Pepadun**

DOI: 10.23960/pepadun.v6i1.257

```
public function login() {
    $userModel      = new UserModel();
    $email          = $this->request->getVar('email');
    $password       = $this->request->getVar('password');
    $user           = $userModel->where('email', $email)->first();
    if(is_null($user)) {
        return $this->respond(['error' => 'Invalid username or password.'], 401);}
    $pwd_verify = password_verify($password, $user['password']);
    if(!$pwd_verify) {
        return $this-> respond(['error' => 'Invalid username or password.'], 401); }
    $key = getenv('JWT_SECRET');
    $iat = time();
    $exp = $iat + 86400;
    $payload = array(
        "iat"       => $iat,
        "exp"       => $exp,
        "email"     => $user['email'], );
    $token = JWT::encode($payload, $key, 'HS256');
    $response = ['message' => 'Login Succesful', 'token' => $token];
    return $this->respond($response, 200); }
```

Code 1. Login function.

```
public function get_kinerja($kode = null, $fakultas = null, $prodi = null, $id_dosen =
null, $tahun = null, $limit = 10, $offset = 0, $sort = 'tahun', $order = 'DESC',
$search = null) {
    $query = $this->db->table('kin_swmp')
          ->join('kegiatan_remun', 'kin_swmp.id_kegiatan = kegiatan_remun.kode')
          ->join('unsur_kegiatan_remun', 'unsur_kegiatan_remun.kode = kegiatan_remun.
            kode_unsur_kegiatan')
          ->join('data_dosen', 'TRIM(data_dosen.nidn) = TRIM(kin_swmp.id_dosen)');
    $query->select('kin_swmp.kode as id,
           kin_swmp.tahun,
           TRIM(kin_swmp.id_dosen) as id_dosen,
           TRIM(data_dosen.nip) as nip,
           TRIM(data_dosen.nama) as nama,
           TRIM(kin_swmp.judul_kegiatan) as kegiatan');
    // query option
    if(!empty($kode)){ $query->where('unsur_kegiatan_remun.kode_bidang', $kode); }
    if(!empty($fakultas)){ $query->where('TRIM(data_dosen.fakultas) =',
    TRIM($fakultas)); }
    if(!empty($prodi)){ $query->where('TRIM(data_dosen.prodi) =', TRIM($prodi)); }
    if(!empty($id_dosen)){ $query->where('TRIM(kin_swmp.id_dosen) =',
    TRIM($id_dosen)); }
    if(!empty($tahun)){ $query->where('kin_swmp.tahun', $tahun); }
    if(!empty($search)) {
    $search = strtolower($search);
    $query->groupStart()
           ->like('LOWER(TRIM(data_dosen.nama))', $search)
           ->orlike('LOWER(TRIM(kin_swmp.judul_kegiatan))', $search)
           ->groupEnd(); }
    if (in_array(strtolower($order), ['asc', 'desc'])) { $query->orderBy($sort,
    strtoupper($order)); }
    // limit and offset
    $query->limit($limit, $offset);
    $result = $query->get();
    return $result->getResultArray();}
public function get_detil_kinerja($id = null){
    $query = $this->db->table('kin_swmp')
           ->where('kode', TRIM($id))
           ->get();
    return $query->getResultArray();}
```

Code 2. Get performance of lecturer function.

Vol. 6 No. 1, 2025, pp. 71-80
Jurnal Pepadun

DOI: 10.23960/pepadun.v6i1.257

3.5 API Testing

API testing is used to ensure that the endpoint that has been developed can return the appropriate response. Testing is done using the Postman application which can be seen in Figure 3. The results of API testing of all endpoints that have been developed in this study can be seen in Table 2.

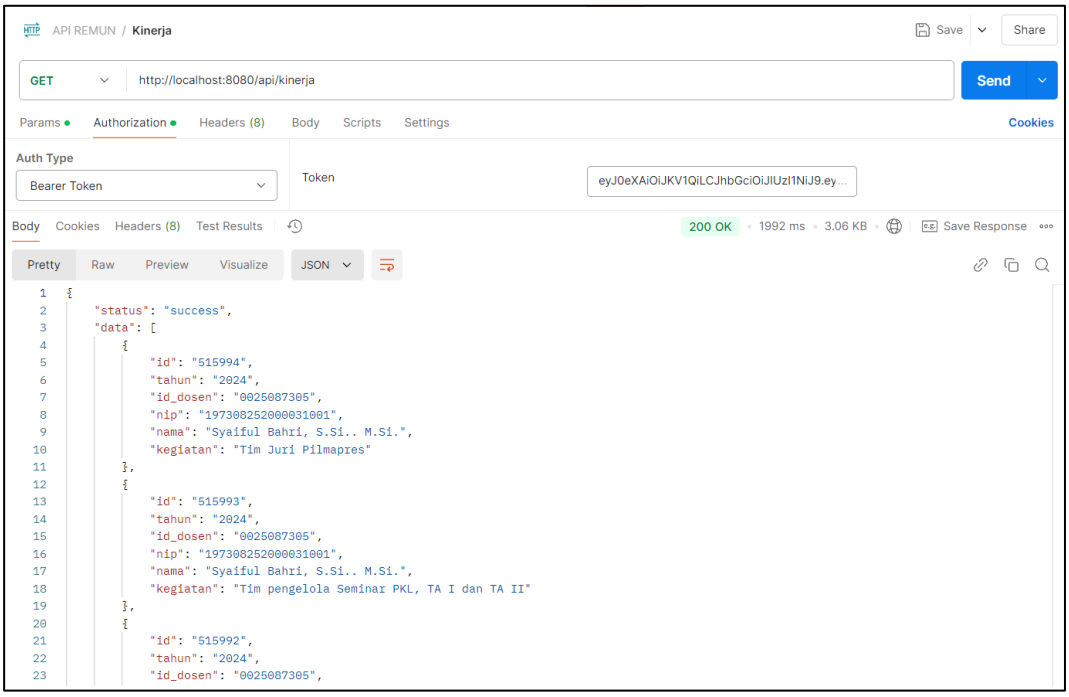

Figure 3. API testing gets performance data.

Table 2. API test results.

No	Method	Endpoint	Status Code	Respon	Avg Response Time (ms)	Avg Size (KB)
1	POST	/api/login	200	Generates JWT tokens.	349	0.648
2	GET	/api/pendidikan	200	Returning educational performance data.	708	3.85
3	GET	/api/pelaksanaan_pendidikan	200	Returning performance data in the field of education implementation.	1296	3.83
4	GET	/api/penelitian	200	Returning research field performance data.	706	3.69
5	GET	/api/pengabdian	200	Returning data on service performance.	824	3.47
6	GET	/api/penunjang	200	Returning supporting field performance data.	993	2.98
7	GET	/api/kinerja	200	Returning lecturer performance data.	1534	3.26
8	GET	/api/detil_kinerja	200	Returning detailed lecturer performance data.	888	5.27

DOI: 10.23960/pepadun.v6i1.257

Based on the API testing results in Table 2, it can be seen that the development of the RESTful API of the Unila lecturer remuneration system has successfully responded according to the endpoints that have been determined at the design and implementation stages. The test was performed ten times for each endpoint, and the response time and data size were recorded from each repetition.

The final response time was obtained by calculating the average of all repetitions to ensure an accurate reflection of the system's performance. The results show that the /api/kinerja endpoint using the GET method had the longest average response time of 1534 ms, while the /api/login endpoint using the POST method had the fastest at 349 ms. The variation in response time is influenced by factors such as the volume of data in the database tables, server performance, and the complexity of the queries used during data retrieval.

In terms of response size, the /api/login endpoint produced the smallest average payload at 0.648 KB, while the /api/detil_kinerja endpoint returned the largest at 5.27 KB. This variation reflects the differences in the complexity and amount of data being processed and returned by each endpoint. Overall, the API demonstrates efficient handling of data with acceptable payload sizes across all tested endpoints.

CONCLUSION

This research successfully developed a RESTful API with JWT authentication for the Lecturer Remuneration System at the University of Lampung. The API testing results indicate that all endpoints responded successfully with HTTP 200 status codes, demonstrating stable and reliable functionality. Response times ranged from 349 ms to 1534 ms, with /api/login being the fastest and /api/kinerja the slowest. Response sizes varied between 0.648 KB and 5.27 KB, with the login endpoint returning the smallest payload and the detailed lecturer performance endpoint the largest. These results confirm that the developed API delivers data efficiently and operates with acceptable latency, supporting its suitability for academic performance-related services.

LITERATURE

[1] Hanny, S. Samsugi, and A. Sulistiyawati, "Rancang Bangun Sistem Informasi Pendataan Calon Penerima Bantuan Sosial dan Desa Berbasis Web (Studi Kasus: Desa Cilimus)," *Jurnal Teknologi dan Sistem Informasi*, vol. 4, no. 3, hlmn. 328–339, 2023. doi: https://doi.org/10.33365/jtsi.

[2] J. S. Utama and A. D. Indriyanti, "Pengamanan Restful API Web Service Menggunakan Json Web Token (Studi Kasus: Aplikasi Siakadu Mobile Unesa)," *Journal of Emerging Information Systems and Business Intelligence*, vol. 04, no. 01, hlmn. 8–17, 2023.

[3] I. A. Permana, "Analisis Penilaian Kinerja Dosen Menggunakan Metode Balance Scorecard (Studi Kasus Stt Sangkakala)," *Jurnal Riset Ekonomi dan Bisnis*, vol. 13, no. 2, hlm. 89–99, 2020. doi: http://journals.usm.ac.id/index.php/jreb

[4] M. Nugraha and M. Rosmeida, "Perancangan Sistem Informasi Beban Kerja Dosen Berbasis Web dengan UML," *Jurnal Algoritma*, vol. 19, no. 1, hlm. 141–156, 2021.

[5] M. Iqbal and N. Nurwati, "Penerapan sistem terintegrasi menggunakan RESTful API pada Dealer Management System Panca Niaga Sei Piring," *Jurnal Sistem dan Sains Rekayasa (JSSR)*, vol. 6, no. 1, pp. 219–224, 2023. [Online]. Available: https://doi.org/10.54314/jssr.v6i1.1161

[6] F. Shofyan and R. T. Shita, "Implementasi Web Service Restful API dengan Autentikasi Personal Access Tokens dan Algoritma AES 256," *Jurnal Technology of Information and Communication*, vol. 12, no. 3, pp. 108–114, Mei 2024.

[7] S. A. B. Cahyono, E. R. Kusuma, and A. S. Putra, "Rancangan Pembuatan API Website Data Tanaman Obat dan Langka Kabupaten Kediri," *Jurnal Bulletin of Information Technology*, vol. 3, no. 4, pp. 255–260, Desember 2022.

DOI: 10.23960/pepadun.v6i1.257

[8] F. Shofyan and R. T. Shita, "Implementasi Web Service Restful API dengan Autentikasi Personal Access Tokens dan Algoritma AES 256," *Jurnal Technology of Information and Communication*, vol. 12, no. 3, hlm. 108-114, 2024.

[9] I. Kurniawan, Humaira, and F. Rozi, "Rest API Menggunakan NodeJS pada Aplikasi Transaksi Jasa Elektronik Berbasis Android," *Jurnal Ilmiah Teknologi Sistem Informasi*, vol. 1, no. 4, hlm. 127–132, 2020. doi: http://jurnal-itsi.org

[10] S. A. B. Cahyono, "Rancangan Pembuatan API Website Data Tanaman Obat dan Langka Kabupaten Kediri," *Jurnal Bulletin of Information Technology*, vol. 3, no. 4, hlm. 255–260, Des 2022, doi: 10.47065/bit.v3i1.

[11] Hasanuddin, H. Asgar, and B. Hartono, "Rancang Bangun Rest API Aplikasi Weshare Sebagai Upaya Mempermudah Pelayanan Donasi Kemanusiaan," *Jurnal Informatika Teknologi dan Sains*, vol. 4, no. 1, hlm. 8–14, Feb 2022.

[12] A. Y. Nashikhuddin, J. Karaman, and Y. Litanianda, "Implementasi Api Restful dengan JWT pada Aplikasi E-Commerce Thrifty Shop untuk Otentikasi dan Otorisasi Pengguna," *Jurnal Manajemen Informatika dan Komputerisasi Akuntansi*, vol. 7, no. 2, hlm. 239–246, Okt 2023, doi: 10.46880/jmika.Vol7No2.pp239-246.

[13] Y. L. Pattinama, Ferdiansyah, I. Susanti, and Painem, "Implementasi Rest API Web Service Dengan Otentikasi JSON Web Token Untuk Aplikasi Properti," *Jurnal Informatik*, vol. 19, no. 1, hlm. 81–89, Apr 2023.

[14] G. Y. Gustiegan and P. Painem, "Implementation of a RESTful Web Service with JSON Web Token Authentication and AES-256 Cryptographic Algorithm for Mobile-Based Laboratory Loan Applications at Budi Luhur University," *BIT (Bina InformaTek)*, vol. 19, no. 1, 2022, doi: 10.36080/bit.v19i1.1835.