






# Implementation of Representational State Transfer Application Programming Interface with Bearer Authentication in Business Service Systems

Ni Nyoman Harini Puspita<sup>1</sup>, I Putu Oka Wisnawa<sup>2</sup>,  
I Putu Bagus Arya Pradnyana<sup>3</sup>, and Gde Brahupadhya Subiksa<sup>4</sup>

<sup>1,2,3,4</sup> Information Technology Department, Politeknik Negeri Bali, Bali, Indonesia  
mangari@pnb.ac.id

**Abstract.** This study aims to implement a REST API with Bearer Authentication in a Business Service System, focusing on enhancing security, flexibility, and efficiency in data exchange. The long-term goal is to create a secure and authenticated environment for external clients while facilitating quick and efficient access to available business services. To achieve this objective, the study sets several specific targets. First, a system requirements analysis will be conducted to thoroughly understand the functional and non-functional requirements of the system, as well as stakeholder expectations. Next, an API architecture design that prioritizes security will be carried out, integrating Bearer authentication mechanisms to securely validate each client request. The prototype development will involve the practical implementation of this design, enabling direct testing and initial evaluation of the system's security and performance. The subsequent step is to conduct trials on the prototype, simulating real-world scenarios to test the API's security, interoperability, and performance. Performance evaluation will be the final stage, where the system's overall performance will be assessed to ensure that the implementation meets the expected standards. The method used will involve collaboration between the development team, users, and stakeholders, ensuring that business needs and system security remain the primary focus at every step. By undertaking these steps, this study hopes to make a significant contribution to improving the security, flexibility, and efficiency of the business service system, creating a reliable environment for critical business data exchange.

**Keywords:** Authentication, Bearer Token, Rest API, Response Time, Throughput

## 1 Introduction

System integration is a crucial aspect of improving organizational efficiency and productivity. The importance of interoperability in Information Systems through Application Programming Interfaces (APIs) has become a fundamental basis for facilitating inter-system connections (Sjarov et al., 2020). APIs, particularly Representational State Transfer (REST API), can serve as a reliable solution to support

© The Author(s) 2024

A. A. N. G. Saptaka et al. (eds.), *Proceedings of the International Conference on Sustainable Green Tourism Applied Science - Engineering Applied Science 2024 (ICoSTAS-EAS 2024)*, Advances in Engineering Research 249, [https://doi.org/10.2991/978-94-6463-587-4\\_4](https://doi.org/10.2991/978-94-6463-587-4_4)

system interoperability (Yanti & Rihyanti, 2021). The successful implementation of APIs not only enables smooth data exchange but also supports the flexibility and scalability needed to adapt to the dynamic changes in organizational needs (Felicio et al., 2023). Interoperability can be achieved through various methods tailored to specific requirements (Sakowski et al., 2019). At Politeknik Negeri Bali, there are currently several systems that are not yet fully integrated. This creates obstacles to operational efficiency and reduces the capability for quick decision-making. The campus community is currently striving towards secure system integration to minimize disparities and enhance inter-application connectivity. This process is a strategic step in embracing an increasingly sophisticated digitalization era, where the need for integrated information becomes more urgent. This research aims to integrate business services at Politeknik Negeri Bali with the remuneration application by utilizing REST API. The main focus of the study is to transmit revenue data from business services to the remuneration application with optimal speed and security. The advantage of REST API integration lies in its simple architecture, easy understanding, and support for HTTP protocols, allowing for efficient and reliable data exchange (Ong et al., 2015). With this integration, it is expected that the campus can optimize its business and administrative processes, provide accurate data, and improve user experience. This research has a specific objective to integrate business services at Politeknik Negeri Bali with the remuneration application using Representational State Transfer (REST) API. In this context, the use of REST API is implemented with Bearer Authentication, a security mechanism that underpins authentication in the exchange of information between related systems (Adinta et al., 2024). The main advantage of REST API with Bearer Authentication lies in its ability to provide a high level of security through a reliable authentication process (Sopingi et al., 2021). With Bearer Authentication, the integration between business services and the remuneration application becomes more secure, ensuring that only authorized parties can access and transmit data. The use of tokens as authentication keys provides an extra layer of security, preventing unauthorized access and providing a traceable audit trail for each data exchange. This mechanism is crucial, especially in a campus environment with high sensitivity toward personnel and financial data (Adinta et al., 2024). Moreover, REST API with Bearer Authentication Token also supports a more efficient integration process. This system allows relevant parties to access business services and the remuneration application without the need to repeatedly provide authentication information, optimizing system performance and reducing the administrative burden related to authentication (Iqbal & Royal, 2023). Therefore, the use of REST API with Bearer Authentication in this research is expected not only to enhance data security but also to speed up and simplify the integration process at Politeknik Negeri Bali. The results of this study show that the integration of business service and remuneration systems using REST API can support more effective and efficient processes at Politeknik Negeri Bali. The automatic integration of revenue data reduces the risk of human error (Pamungkas et al., 2019). Additionally, the integrated system also facilitates better monitoring of the campus's financial and administrative performance. This integration not only enhances productivity but also creates a solid foundation for the further development of information technology in the future.

## 2 Literature Review

### 2.1 REST API

Data integration plays a highly significant role as a foundation in the development of digital infrastructure in this era. In the context of information technology, the need for integrated data has become a critical factor in determining the effectiveness and efficiency of decision-making processes within an organization. On the other hand, interoperability characterizes the ability of systems to communicate, share data, and operate together with other systems without requiring additional adjustments (Sambari & Santika, 2024). In 2021, Muhammad Faisal conducted research that demonstrated how interoperability allows information delivery across different platforms to work efficiently, thereby enhancing the effectiveness and efficiency of academic processes. Another advantage is that it simplifies the maintenance and development processes of the system. Interoperability implementation was also conducted by Martin Sjarov and his team in 2020, emphasizing that initial decisions in product development have a significant impact on the performance, quality, and total cost of the product. The primary focus was on Systems Engineering and the concept of Industry 4.0, introducing a new principle called “Design for Interoperability” (DfIOp). This principle aims to meet product needs in the technical planning phase, opening up opportunities for further developments such as “Design for Data Analytics” and “Design for Digital Twin” (Sjarov et al., 2020). The results of this research show that the implementation of interoperability in software engineering is highly beneficial.

REST API became popular and widely adopted in software development following the publication of Fielding's dissertation. The advantages of REST, such as its simple design, scalability, and flexibility, have made it a popular choice for developing application programming interfaces (APIs) in web environments (Grembowski et al., 2002). As the web has grown and there has been a shift towards service-oriented architecture, REST API has increasingly become the de facto standard in web and mobile application development. The use of REST API continues to grow in line with the increasing complexity of applications, the need for cross-platform interactions, and the demand for easier integration between various systems (Laipaka, 2022).

### 2.2 Bearer Authentication

Recent developments in the field of information technology, particularly in REST API development, have shown a significant increase in the implementation of more advanced security methods. One of the leading innovations is the use of Bearer Authentication to secure REST APIs (Yanti & Rihyanti, 2021). This method allows the use of a specific token called a “Bearer Token” as a form of authentication. The Bearer Token functions as an identification token that clients must possess to access protected resources on the server. With this method, developers can implement a stronger security layer, as the token is only issued to legitimate entities with the appropriate access rights. The main advantage of Bearer Authentication lies in its flexibility, where users can include the token in every HTTP request without needing to store credential information on the server. Thus, REST APIs with Bearer Authentication not only

enhance security levels but also provide efficiency in the authentication and authorization process for accessing protected resources. This development reflects a positive evolution in delivering reliable security solutions that meet the demands of the complexity of modern application systems.

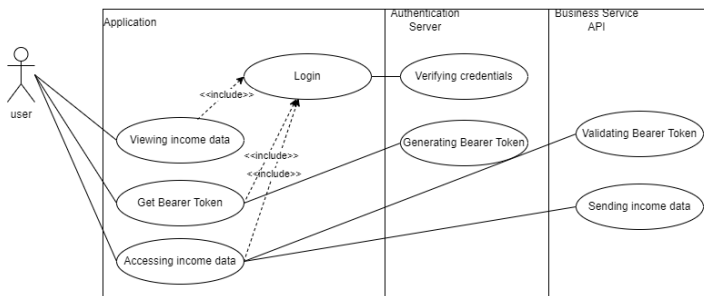
### 3 Methodology

#### 3.1 Research Method

This research was conducted at the Politeknik Negeri Bali campus in Jimbaran from April 2024 to November 2024. The object of the research is the development and implementation of a REST API using Bearer Authentication on a Business Service System connected to a remuneration system. The main focus of this research is on the integration and interaction between the business services as the data source and the remuneration system for retrieving income data. During the study, the research will involve the design, development, and testing of the REST API implementation that utilizes Bearer Authentication security mechanisms, to ensure security and authentication in the data exchange between the two systems.

#### 3.2 Design System

**Use Case.** Create a use case diagram that illustrates the integration architecture between the Business Service System and the Remuneration Application using REST API with Bearer Token. This diagram should show the involved actors, relevant use cases, and the relationships between the use cases.



**Figure 1.** Use case

**Diagram Sequence.** Only two levels of headings should be numbered. Lower-level headings remain unnumbered; they are formatted as run-in headings, as shown in Figure 2.

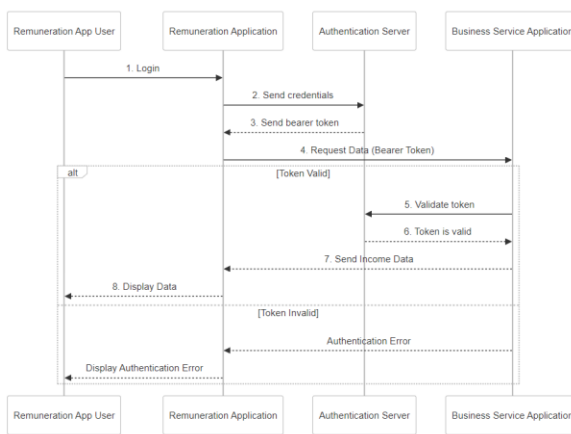


Figure 2. Diagram sequence

**Class Diagram.** The class diagram can be seen in Figure 3.

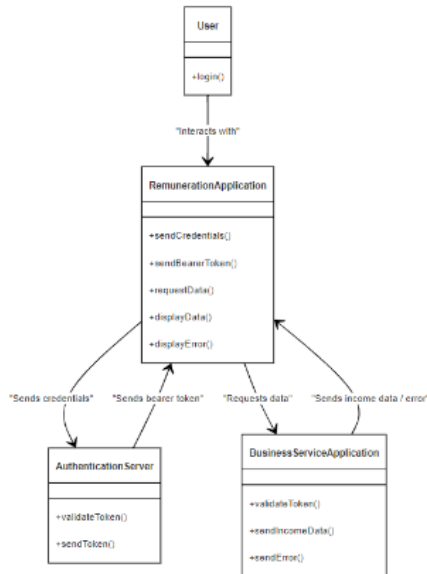


Figure 3. Class diagram

**Mind Map.** The Mind Map can be seen in Figure 4.

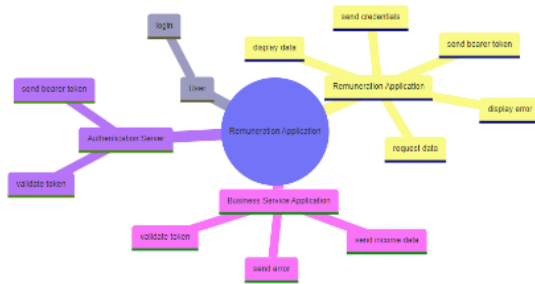


Figure 4. Mind map

## 4 Result and Discussion

### 4.1 Implementation of REST API Bearer Authentication

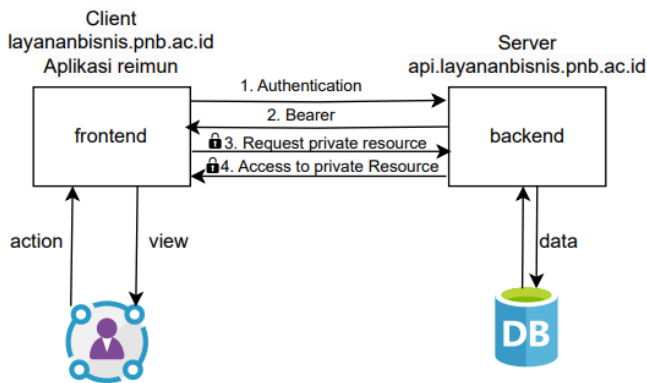


Figure 5. Mechanism of system work

The working mechanism of the diagram above is described as follows.

**User Authentication.** The user authenticates through the frontend interface by entering their credentials (e.g., username and password). After the user submits this information, the front end sends a request to the API in the backend to verify the credentials.

**Sending Bearer Token.** Once the backend verifies the user's credentials and confirms the user is valid, it sends a bearer token back to the frontend. This token serves as proof of the user's identity and is used to access protected resources in the backend.

**Sending Private Requests.** With the received bearer token, the user can now send private requests to the backend. These requests may include accessing specific data or features that require authorization. The front end includes the bearer token in the request header to ensure the backend can verify the user's identity.

**Accessing Private Resources.** The backend receives the request containing the bearer token and validates the token. If the token is valid, the backend grants access to the requested private resources and sends a response back to the frontend.

**Further Interaction.** After receiving the response from the backend, the frontend can display the requested data to the user or perform further actions based on the response.

Here is the API script that is used.

```
public function login(Request $request){
    $validator = Validator::make($request->all(), [
        'email' => ['required', 'string', 'email'],
        'password' => ['required', 'string'],
    ]);
    if ($validator->fails()) {
        return $this->responseError('Login failed', 422,
        $validator->errors());
    }
    if (Auth::attempt(['email' => $request->email, 'password'
=> $request->password])) {
        $user = Auth::user();
        $response = [
            'token' => $user->createToken('MyToken')-
            >accessToken,
            'first_name' => $user->first_name,
            'last_name' => $user->last_name,];
        return $this->responseOk($response);
    } else {
        return $this->responseError('Wrong email or
password', 401);
    }
}
```

Here is the step-by-step explanation:

1. Input Validation:

- The function starts by validating the incoming request data using the Validator class.
- It checks that the email field is required, is a string, and follows a valid email format.

- It also ensures that the password field is required and is a string.
  - If the validation fails, the function returns an error response with a status code of 422, along with the validation errors.
2. Authentication Attempt:
    - If validation passes, the function then attempts to authenticate the user using the Auth::attempt() method.
    - It checks the provided email and password against the stored credentials.
  3. Successful Authentication:
    - If the authentication is successful, the function retrieves the authenticated user using Auth::user().
    - It then creates an access token for the user using the create token method and prepares a response that includes:
      - The generated token.
      - The user's first name and last name.
    - Finally, it returns a successful response with a status code of 200, along with the user information and token.
  4. Failed Authentication:
    - If authentication fails (due to incorrect email or password), the function returns an error response with a status code of 401 and a message indicating that the credentials are incorrect.

The login feature is a strong example of user authentication in action. It starts by confirming that the email format and password are entered correctly. It is to authenticate the user with the supplied credentials after successful validation. If authentication is successful, an access token and the user's first and last names are generated and returned in a structured response. On the other hand, it displays an appropriate error message if authentication fails. By providing unambiguous feedback depending on the authentication result, this method not only improves user experience but also strengthens security through input validation and token generation.

Figures 6 and 7 are the images of testing of API using the Bearer token.

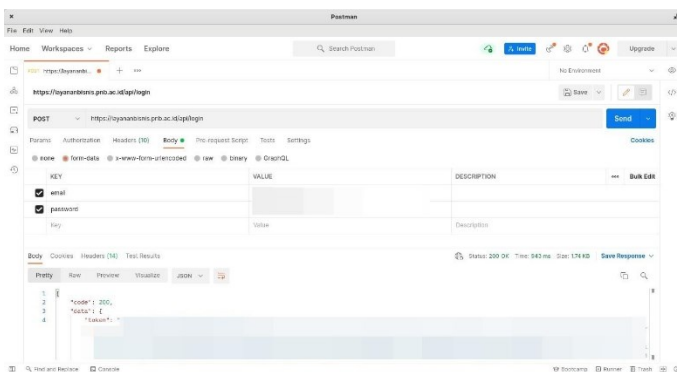


Figure 6. Setting postman



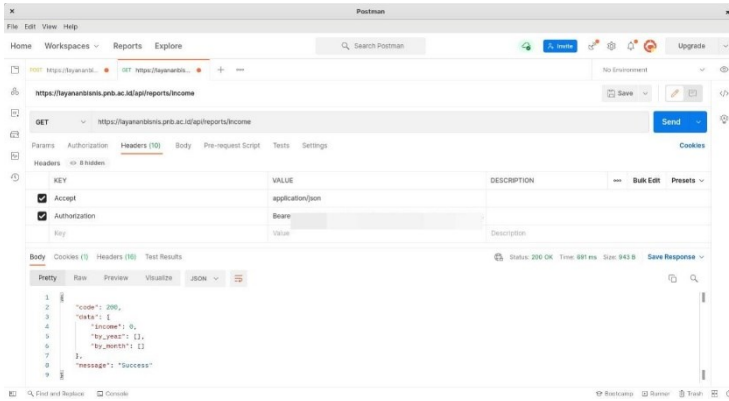


Figure 7. Testing data access through API

Integration testing through Postman has been completed, with all API requests responding as expected, indicating that the integration process is smooth and valid. Based on the performance testing analysis with a simulation of 50 concurrent users, all test scenarios were executed as planned. The average response time for each endpoint is under 2 seconds, with stable throughput and no significant increase in error rates. Server resource usage also remains within reasonable limits, showing that the API can handle high loads effectively, meeting all predefined success criteria.

## 5 Conclusion

API performance testing with a simulation of 50 concurrent users successfully met all success criteria, showing fast response times, stable throughput, and very low error rates. This result indicates that the API has been well-implemented and is ready for real-world use, with the ability to handle high loads without significant performance degradation.

## References

Adinta, B., Winarsih, W. & Ningsih, S. (2024). Implementasi restful API pada sistem pemesanan dan pengiriman makanan. *Jurnal Sistem Informasi Bisnis*, 5(2), 122–129. <https://doi.org/10.55122/junsibi.v5i2.1203>.

Felicio, D., Simao, J., & Datia, N. (2023). RapiTest: Continuous Black-Box testing of restful web APIs. *Procedia Computer Science*, 219, 537–545. <https://doi.org/10.1016/j.procs.2023.01.322>.

Grembowski, T., Lien, R., Gaj, K., Nguyen, N., Bellows, P., Flidr, J., Lehman, T., & Schott, B. (2002). Comparative analysis of the hardware implementations of hash functions SHA-1 and SHA-512. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in*

- Artificial Intelligence and Lecture Notes in Bioinformatics*), 2433, 75–89. [https://doi.org/10.1007/3-540-45811-5\\_6](https://doi.org/10.1007/3-540-45811-5_6).
- Iqbal, M., & Royal, S. (2023). Penerapan sistem terintegrasi menggunakan restful API pada dealer management system panca niaga sei piring. In *Journal of Science and Social Research* (Issue 1). <http://jurnal.goretanpena.com/index.php/JSSR>.
- Laipaka, R. (2022). Penerapan JWT untuk Authentication dan Authorization pada Laravel 9 menggunakan Thunder Client. *Seminar Nasional Corisindo*.
- Pamungkas, Y., Santoso, A. B., Ashari, B., Sensuse, D. I., Mishbah, M., & Meiyanti, R. (2019). Evaluation of interoperability maturity level: Case study Indonesian directorate general of tax. *Procedia Computer Science*, 157, 543–551. <https://doi.org/10.1016/j.procs.2019.09.012>.
- Sakowski, A., Dangelmaier, M., Hertwig, M., & Spath, D. (2019). Bidirectional interoperability of product engineering and manufacturing enhancing mass customization. *Procedia Manufacturing*, 39, 81–89. <https://doi.org/10.1016/j.promfg.2020.01.231>.
- Sambari, P. A., & Santika, R. R. (2024). Implementasi web service pada aplikasi presensi menggunakan metode restful API pada PT. CDC. *4 th Seminar Nasional Mahasiswa Fakultas Teknologi Informasi (SENAFTI)*, 3(1).
- Sopingi, S., Nastiti, F. E., & Majid, A. S. (2021). Implementasi JSON Web Token Authentication pada Aplikasi Pembayaran Berbasis Mobile. *Prosiding Seminar Nasional Hukum, Bisnis, Sains Dan Teknologi*, 2(1), 343.
- Ong, S. P., Cholia, S., Jain, A., Brafman, M., Gunter, D., Ceder, G. Persson, K.A. (2015). The materials Application Programming Interface (API): A simple, flexible and efficient API for materials data based on Representational State Transfer (REST) principles. *Computational Materials Science*, 97, 209–215.
- Sjarov, M., Kibkalt, D., Lechler, T., Selmaier, A., & Franke, J. (2020). Towards “design for Interoperability” in the context of Systems Engineering. *Procedia CIRP*, 96, 145–150. <https://doi.org/10.1016/j.procir.2021.01.067>.
- Yanti, S. N., & Rihyanti, E. (2021). Penerapan rest API untuk sistem informasi film secara daring. *Jurnal Informatika Universitas Pamulang*, 6(1), 195. <https://doi.org/10.32493/informatika.v6i1.10033>.

**Open Access** This chapter is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License (<http://creativecommons.org/licenses/by-nc/4.0/>), which permits any noncommercial use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

