Integrating Travel Booking Management System Using Rest-API

Faiza Renaldi, Hary Suryanto, Agya Java Maulidin, Irma Santikarama
Department of Informatics, Universitas Jenderal Achmad Yani, Indonesia
faiza.renaldi@unjani.ac.id, harysuryanto@student.unjani.ac.id,
java.maulidin@student.unjani.ac.id, irma.santikarama@lecture.unjani.ac.id

Abdul Talib Bon
Department of Production and Operations, University Tun Hussein Onn Malaysia, Malaysia
talibon@gmail.com

Abstract

Many traveling arrangements can be made online at this time, such as lodging, transportation, and choosing tourist attractions. Studies on online tourism systems discuss a lot about booking service systems at hotel reservations, transportation reservations, or booking tour tickets. However, most of these studies were carried out only on individual processes and were not integrated. This study built a web service for travel package booking that allows travelers to book transportation, hotels, and travel tickets in one transaction. The use of API technology combines parameters from ordering a travel package, then carrying out a comprehensive calculation. In the end, travelers will get many options to combine their travel packages automatically. In this study, the validity of the API was tested. Testing the API built resulted in an average time of 2946 ms from 50 trials of API calls by ten different users accessing the system simultaneously. We suggest adding API connections with providers of travel location review information so that more details will be available in one particular place.

Keyword
API-based system, travel package, order management system

1. INTRODUCTION

Web services are software systems designed to support machine-to-machine interactions that operated over a network (Chen et al. 2017). The web service has an interface described in a machine-processable format (Paik et al. 2017). In this modern era, web services have been used in various fields such as e-commerce (Sohaib and Kang 2017), online payment (Listyani et al. 2016), and data provider (Perwira and Santosa 2017). Web service is a system built to manage data or perform an operation based on the requests. Then the client retrieves the result of the process (Jianxiao Liu et al. 2016). We can find this thing in the microservices concept. The advantage of using a web service is that application developers do not need to create a difficult function to perform if there is already a web service that can do similar work. For example, when a group of developers wants to do a payment module for an application they are developing, they can take advantage of a web service that can handle payment transactions such as Midtrans. There are two types of web services that are widely used, namely, SOAP (Simple Object Access Protocol) and REST (Representational State Transfer) (Rafalimanana et al. 2020). SOAP is an XML based protocol, while REST is an architectural based protocol. Also, the SOAP protocol is HTTP and SMTP, and FTP, while REST is HTTP. So far, web services are widely used as a backend in microservice systems. It provides the advantages of good system modularity, easy maintenance, and good scalability. Thus, web services are used as a system capable of handling specific tasks based on client requests so that the software on the client can have a good performance.

One web service that is widely used is Rest-API (Hu et al. 2017). The Rest-API is very popular because it is easy to use, easy to understand, and has better performance (Neumann et al. 2018). Apart from that, the Rest-API also allows sending data using a lightweight data format such as JSON (Barbaglia et al. 2016), which allows for easier parsing (Jian Liu et al. 2019). The Rest-API uses the HTTP methodology defined by the RFC 2616 protocol, like: GET to retrieve resources; PUT to modify resources; POST to create a new resource; and DELETE to delete resources (Putra and Putera 2019). To run the Rest-API, the client needs to run a URL that points to a designated method. This process is known as a request. Then, the client will receive data. This data is known as a response. One example of its use is a weather application that gets data about the weather from a web service via an API.

Rest-API is very popular, and many developers use it, one of which is in the tourism industry (Anugrah et al. 2019). The Rest-API is used to handle hotel bookings (Rauf et al. 2018) (Parra and María 2017) such as in Booking.com application, transportation booking (Jnr 2020) (Chemutai 2019) such as in Tiket.com application, dan booking tour tickets (Rachmaniah et al. 2018) (Garcia et al. 2018) such as in Traveloka application. These applications provide evidence that using Rest-API can provide benefits. One of them is in the travel industry. In hotel reservations, the Rest-API is used to display hotel lists, hotel searches, hotel prices, hotel availability status, hotel reservations, and others. In transportation bookings, Rest-API is used to view the list of available airlines and make ticket bookings. Rest-API is used to see available tourist attractions along with their schedules and prices in booking tourist tickets. Although there have been many uses of the Rest-API in the tourism industry (Rachmaniah et al. 2018), but until now there has been no use of the Rest-API (Nesi et al. 2016) to integrate hotel, transportation, and tour ticket booking transactions. Integrated hotel reservations, transportation, and tourist tickets can make it easier for end-users to book travel packages. This research was conducted to answer the existing gaps and build a web service in the form of a Rest-API that integrates hotel booking, transportation, and tour tickets so that they can be done in one transaction.

2. RESEARCH METHOD

There are five steps in this research, as shown in Figure 1.

Figure 1. Research Method

Different developers create APIs on each system, so they have different ways of using them. The first step in this research method is to analyze the hotel reservation system's API, transportation reservations, and tour ticket reservations to find out how it works and how to use each API. The results of the analysis are then used to determine the structure of the web service. The next step is to develop a web service. There is a line of code in the developed web service that functions to integrate the hotel reservation system's API, the transportation reservation system, and the tour ticket booking system. In the next step, we create a Rest-API so that the Rest server can communicate with the Rest client. Testing is carried out at the last step to measure the success rate of the system. The final step also describes the results of the research at each step.

2.1 Analyzing the API Structure of the Hotel Booking System, Transportation Booking, and Tour Ticket Booking

Analysis of the API structure is needed to find out the design of the existing API. The investigation is carried out to determine how it works and find out the use of existing APIs. The API used in this study comes from API providers (Booking.com provides the API for hotel bookings, Tiket.com provides API for transportation booking, and Traveloka.com provides API for tour ticket booking). Each of these APIs has a different design, as shown in Tables 1, 2, and 3.

Table 1. Hotel Booking API

Function	Message Type	Message Name	Fill in the Message/Attribute
Show Hotel List	Input	listRequest	- location
	Output	listResponse	- hotel_id
			- hotel_name
			- location
			- rate
			- price
			- availability
			- hotel_img_url
Add Order	Input	addOrderRequest	- hotel_id
			- checkin
			- checkout

			- jumlah_tamu_dewasa - jumlah_tamu_anak
	Output	addOrderResponse	- order_id - message
Order Detail	Input	orderDetailRequest	- order_id
	Output	orderDetailResponse	- order_id - hotel_id - checkin - checkout - jumlah_tamu_dewasa - jumlah_tamu_anak

The Show Hotel List function has a message listRequest and listResponse. Message listRequest is the data sent when requesting the server to request hotel list data based on location. Message list Response is the result of making the request, namely in the form of the requested hotel list data. The Add Order function has the addOrderRequest and addOrderResponse messages. The addOrderRequest message is data sent when requesting the server to ask the server to add new order data. The addOrderResponse message is the result of a request, which is the success status of adding orders. The Order Detail function has a message orderDetailRequest and orderDetailResponse. OrderDetailRequest message is data sent when requesting the server to request order detail data based on order_id. OrderDetailResponse message results from making the request, which is detailed data for the requested order. Figures 2, 3, and 4 show the outputs of Show Hotel List, Add Order, and Order Detail functions.


```
[
  {
    "hotel_id": 4362459,
    "hotel_name": "The Oberoi Lombok",
    "location": "Lombok, Indonesia",
    "rate": "4.5/5",
    "price": 4200000,
    "availability": "available",
    "hotel_img_url": "https://www.oberoihotels.com/-/media/oberoi-hotels
      /website-images/the-oberoi-lombok--luxury-pavilion-ocean-view.jpg"
  }
]
```


Figure 2. The output of Function Show Hotel List


```
{
  "order_id": 2057394,
  "message": "order success"
}
```


Figure 3. The output of Function Add Order


```
{
  "order_id": 2057394,
  "hotel_id": 4362459,
  "checkin": "2020-10-01",
  "checkout": "2020-10-03",
  "jumlah_tamu_dewasa": 2,
  "jumlah_tamu_anak": 0
}
```


Figure 4. The output of Function Order Detail

Table 2. Transportation Booking API

Function	Message Type	Message Name	Fill in the Message/Attribute
Show Airlines List	Input	listRequest	- departure_airport - arrival_airport - checkin - checkout
	Output	listResponse	- airline_id - airline_name - airline_logo_url - price
Add Order	Input	addOrderRequest	- departure_airport

			- arrival_airport - checkin - checkout - kelas_kabin - nama_lengkap_pemesan - email_pemesan - no_hp_pemesan
	Output	addOrderResponse	- order_id - message
Order Detail	Input	orderDetailRequest	- order_id
	Output	orderDetailResponse	- airline_id - airline_name - airline_logo_url - price - departure_airport - arrival_airport - checkin - checkout - kelas_kabin - nama_lengkap_pemesan - email_pemesan - no_hp_pemesan

The Show Airlines List function has a message listRequest and listResponse. Message list Request is data that is sent when requesting the server to request airline list data based on departure_airport, arrival_airport, check-in, and check-out. Message list Response is the result of making the request, namely in the form of the requested airline list data. The Add Order function has the addOrderRequest and addOrderResponse messages. The addOrderRequest message is data sent when requesting the server to ask the server to add new order data. The addOrderResponse message is the result of a request, which is the success status of adding orders. The Order Detail function has a message orderDetailRequest and orderDetailResponse. OrderDetailRequest message is data sent when requesting the server to request order detail data based on order_id. OrderDetailResponse message results from making the request, which is detailed data for the requested order. Figures 5, 6, and 7 show the outputs of the Show Airlines List, Add Order, and Order Detail functions.


```
[
  {
    "airline_id": 7952454,
    "airline_name": "Garuda Indonesia",
    "airline_logo_url": "https://akcdn.detik.net.id/customthumb/2009/07/23/4
      /Garuda-Logo-Vertical-dalam.jpg",
    "price": 3580000
  }
]
```


Figure 5. The output of Function Show Airlines List


```
{
  "order_id": 7389053,
  "message": "order success"
}
```


Figure 6. The output of Function Add Order

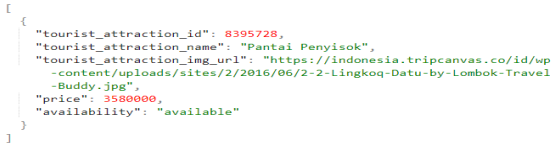

```
{
  "airline_id": 7389053,
  "airline_name": "Garuda Indonesia",
  "airline_logo_url": "https://akcdn.detik.net.id/customthumb/2009/07/23/4
    /Garuda-Logo-Vertical-dalam.jpg",
  "price": 3580000,
  "departure_airport": "Bandar Udara Internasional Husein Sastranegara",
  "arrival_airport": "Bandar Udara Internasional Zainuddin Abdul Madjid",
  "checkin": "2020-09-30",
  "checkout": "2020-10-01",
  "kelas_kabin": "Kelas Ekonomi",
  "nama_lengkap_pemesan": "Justin"
}
```


Figure 7. The output of Function Order Detail

Table 3. Tour Ticket Booking API

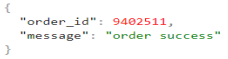
Function	Message Type	Message Name	Fill in the Message/Attribute
Show Tourist Attraction List	Input	listRequest	- tanggal_kunjungan
	Output	listResponse	- tourist_attraction_id - tourist_attraction_name - tourist_attraction_img_url - price - availability
Add Order	Input	addOrderRequest	- tanggal_kunjungan_wisata - jumlah_tamu_dewasa - jumlah_tamu_anak - nama_lengkap_pemesan - no_hp_pemesan - email_pemesan
	Output	addOrderResponse	- order_id - message
Order Detail	Input	orderDetailRequest	- order_id
	Output	orderDetailResponse	- tourist_attraction_id - tourist_attraction_name - tourist_attraction_img_url - price - tanggal_kunjungan - nama_lengkap_pemesan - no_hp_pemesan - email_pemesan - jumlah_tamu_dewasa - jumlah_tamu_anak

The Show Tourist Attraction function has a message listRequest and listResponse. Message listRequest is the data sent when requesting the server to request a list of tourist attractions based on the visit's date. Message list Response is the result of making the request, which is a list of requested tourist attractions. The Add Order function has the addOrderRequest and addOrderResponse messages. The addOrderRequest message is data sent when requesting the server to ask the server to add new order data. The addOrderResponse message is the result of a request, which is the success status of adding orders. The Order Detail function has a message orderDetailRequest and orderDetailResponse. OrderDetailRequest message is data sent when requesting the server to request order detail data based on order_id. OrderDetailResponse message results from making the request, namely in detailed data for the requested order. Figures 8, 9, and 10 show the Show Tourist Attraction List, Add Order, and Order Detail functions output.



```
[
  {
    "tourist_attraction_id": 8395728,
    "tourist_attraction_name": "Pantai Penyisok",
    "tourist_attraction_img_url": "https://indonesia.tripcanvas.co/id/wp
      -content/uploads/sites/2/2016/06/2-2-Lingkoq-Datu-by-Lombok-Travel
      -Buddy.jpg",
    "price": 3580000,
    "availability": "available"
  }
]
```


Figure 8. The output of Function Show Tourist Attraction List



```
{
  "order_id": 9402511,
  "message": "order success"
}
```


Figure 9. The output of Function Add Order

```
{
  "tourist_attraction_id": 9402511,
  "tourist_attraction_name": "Pantai Penyisok",
  "tourist_attraction_img_url": "https://indonesia.tripcanvas.co/id/wp
    -content/uploads/sites/2/2016/06/2-2-Lingkoq-Datu-by-Lombok-Travel-Buddy
    .jpg",
  "price": 600000,
  "tanggal_kunjungan": "2020-10-02",
  "nama_lengkap_pemesan": "Justin",
  "no_hp_pemesan": "085274283312",
  "email_pemesan": "justin@gmail.com",
  "jumlah_tamu_dewasa": 2,
  "jumlah_tamu_anak": 0
}
```


Figure 10. The output of Function Order Detail

2.2 Building a Web Service

Several architectures are widely used in web service development, including SOAP (Simple Object Access Protocol) and REST (REpresentational State Transfer). SOAP architecture produces data output in XML format, while REST architecture produces data output in JSON format. The choice of architecture is determined based on the needs of the web service. Suppose the need for a web service is to have a guaranteed level of reliability and security. In that case, we can use SOAP architecture because it offers additional standards for ensuring processing operations and asynchronous calls. However, if the web service has limited bandwidth and resources, then the REST architecture needs to be chosen. Each architecture has advantages and disadvantages, but in this study, we use the REST architecture to build web services because of its ease of use. The web service works in a way that the client will make a series of calls via a request to the server where the hosted web service is. Requests are made through what is known as remote procedure calls. Also, the REST architecture on web services is very suitable for implementation in mobile-based client applications because the data sent has the JSON format, which has a small size to provide good performance. This study built a web service that integrates several booking transaction functions, including hotel reservations, transportation, and tour tickets. The available functions come from different APIs.

2.3 Building an Integration System

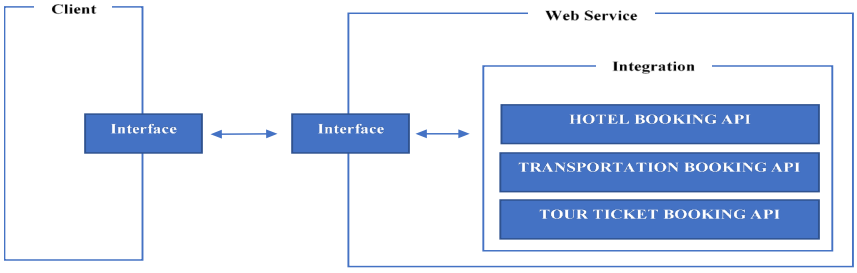


Figure 11. Integration System

Hotel, transportation, and tourist ticket booking transactions are carried out separately and originate from different APIs. In this research, the three transactions are integrated to become one transaction. At this step, we set up a system to integrate the three transactions, shown in Figure 11.

2.4 Building Rest API

The Rest API uses HTTP as a medium for data transmission. Methods in the API are accessed via URL (endpoints). The URL contains the method name along with the parameters it requires. There are several methods in the built Rest API, as shown in Table 4.

Table 4. List of API URLs

No. API	URL	Method	Parameter	Description
API1	/showTravelPackageList	GET	api_key	Used to request data
API2	/showTravelPackageDetail	GET	api_key, package_id	Used to request data based on package_id

API3	/bookTravelPackage	POST	api_key, package_id, hotel_id, hotel_checkin, hotel_checkout, airport_checkin, airport_checkout, jumlah_tamu_dewasa, jumlah_tamu_anak, departure_airport, arrival_airport, nama_lengkap_pemesan, email_pemesan, no_hp_pemesan, tanggal_kunjungan_wisata	Used to book tour packages

The showTravelPackageList method is a method with a GET type that functions to get a list of travel package data, shown in Figure 12. The showTravelPackageDetail method is a method with the GET type, which works to get detailed data from travel packages selected based on package_id, shown in Figure 13. The bookTravelPackage method is a POST type method that functions to make travel package purchases by sending some of the required transaction data, shown in Figure 14. These methods have an api_key parameter containing an authentication code that needs to fill to access that method.


```
[
  {
    "package_id": 3685629,
    "package_name": "Paradise in Lombok",
    "package_description": "Description here...",
    "hotel_name": "The Oberoi Lombok",
    "hotel_img_url": "https://www.oberoihotels.com/-/media/oberoi-hotels
      /website-images/the-oberoi-lombok--luxury-pavilion-ocean-view.jpg",
    "airlines_name": "Garuda Indonesia",
    "airlines_logo_url": "https://akcdn.detik.net.id/customthumb/2009/07/23
      /4/Garuda-Logo-Vertical-dalam.jpg",
    "tourist_attraction_name": "Pantai Penyisok",
    "tourist_attraction_img_url": "https://indonesia.tripcanvas.co/id/wp
      -content/uploads/sites/2/2016/06/2-2-Lingkoq-Datu-by-Lombok-Travel
      -Buddy.jpg"
  }
]
```


Figure 12. The output of Method Show Travel Package List


```
{
  "package_id": 3685629,
  "package_name": "Paradise in Lombok",
  "package_description": "Description here...",
  "hotel_name": "The Oberoi Lombok",
  "hotel_img_url": "https://www.oberoihotels.com/-/media/oberoi-hotels
    /website-images/the-oberoi-lombok--luxury-pavilion-ocean-view.jpg",
  "airlines_name": "Garuda Indonesia",
  "airlines_logo_url": "https://akcdn.detik.net.id/customthumb/2009/07/23/4
    /Garuda-Logo-Vertical-dalam.jpg",
  "tourist_attraction_name": "Pantai Penyisok",
  "tourist_attraction_img_url": "https://indonesia.tripcanvas.co/id/wp
    -content/uploads/sites/2/2016/06/2-2-Lingkoq-Datu-by-Lombok-Travel-Buddy
    .jpg"
}
```


Figure 13. The output of Method Show Travel Package Detail


```
{
  "order_id": 3342585,
  "message": "order success"
}
```


Figure 14. The output of Method Book Travel Package

2.5 Testing

At this step, we test API calls with ten different users accessing the system simultaneously. It is conducted to determine the speed of the API used. We trial them by calculating the time from the beginning of the user saves the order until the order is stored. The system embeds the measuring time in the ordering process.

3. RESULTS AND DISCUSSIONS

This study builds a web service with the REST architecture associated with each required parameter in the API used, namely hotel reservations, transportation, and tour tickets. This study's web service architecture has three main entities, as shown in Figure 15, namely service requester/client, service provider/web service, resource provider. The web service processes the resources obtained from the resource provider and then integrates them, then the client uses the services it needs from the web service.

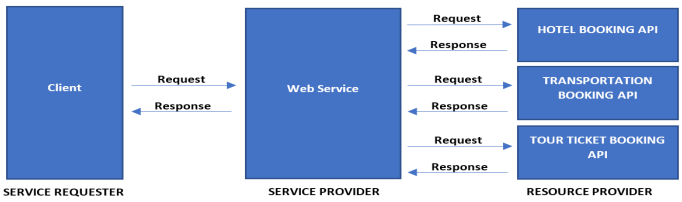


Figure 15. Web Service Architecture

3.1 Web Service

This web service development uses the PHP programming language and the CI framework (Codeigniter), and we deploy it in online hosting. The hotel, transportation, and tour ticket booking process are integrated by this web service, as shown in Table 5.

Table 5. Hotel Booking, Transportation Booking, and Tour Ticket Booking API URLs

API	URL	Method	Parameter	Description
Hotel Booking	https://api.tiket.com /addOrder	POST	hotel_id, checkin, checkout, jumlah_tamu_dewasa, jumlah_tamu_anak	Used to make hotel reservations
Transportation Booking	https://api.booking.com /addOrder	POST	departure_airport, arrival_airport, checkin, checkout, kelas_kabin, nama_lengkap_pemesan, email_pemesan, no_hp_pemesan	Used to make airplane transportation orders
Tour Ticket Booking	https://api.traveloka.com/addOrder	POST	tanggal_kunjungan_wisata, jumlah_tamu_dewasa, jumlah_tamu_anak, nama_lengkap_pemesan, no_hp_pemesan, email_pemesan	Used to make tour ticket reservations

Hotel Booking API is a URL that functions to make hotel reservations, including several parameters, namely hotel_id, check-in, check-out, jumlah_tamu_dewasa jumlah_tamu_anak. The Transportation Booking API is a URL that works to place an airplane transportation order, by including several parameters, namely departure_airport, arrival_airport, check-in, check-out, kelas_kabin, nama_lengkap_pemesan, email_pemesan, and no_hp_pemesan. The Tour Ticket Booking API is a URL that functions to book travel tickets, including several parameters: tanggal_kunjungan_wisata, jumlah_tamu_dewasa, jumlah_tamu_anak, nama_lengkap_pemesan, no_hp_pemesan, and email_pemesan. The three APIs outputs are the same, namely the JSON format data containing the order_id and message key, as shown in Figure 16.


```
{
  "order_id": 9402511,
  "message": "order success"
}
```


Figure 16. The output of API Hotel Booking, Transportation Booking, and Tour Ticket Booking

3.2 Integration

The integration process is carried out by running the APIs described in point A in the web service. Some parameters need to fill in the URL of the APIs, namely hotel_id, hotel_checkin, hotel_checkout, airport_checkin, airport_checkout, jumlah_tamu_dewasa, jumlah_tamu_anak, departure_airport, arrival_airport, nama_lengkap_pemesan, email_pemesan, no_hp_pemesan, tanggal_kunjungan_wisata. These parameters retrieve a value based on input data from the user system (client).

3.3 Rest-API

Rest-API is created for the web service to make user software (client) able to implement it. The Rest-API URL has parameters, as shown in Table 6. When the client runs the URL as a request, the client will retrieve a response in JSON formatted data containing the key, namely the order_id and message, as shown in Figure 17. The order_id key

is the unique data from the recorded transaction data, while the key message is a message that explains whether the transaction was successful or failed.

Table 6. Rest-API URL

API	URL	Method	Parameter	Description
Book Travel	/bookTravelPackage	POST	api_key, package_id, hotel_id, hotel_checkin, hotel_checkout, airport_checkin, airport_checkout, jumlah_tamu_dewasa, jumlah_tamu_anak, departure_airport, arrival_airport, nama_lengkap_pemesan, email_pemesan, no_hp_pemesan, tanggal_kunjungan_wisata	Used to book travel packages

The Book Travel API is a URL that functions to book a travel package. Bookings are made by including several parameters, as shown in Table 6.

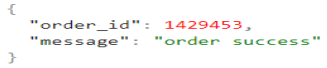


Figure 17. The output of Method bookTravelPackage

3.4 Testing Results

The tests carried out in this study include testing API calls with ten different users accessing the system simultaneously. API call testing is conducted to determine the speed of the API used. We examine the condition by calculating the time from the user's beginning to save the order until the order is stored. The system embeds the measuring time in the ordering process. Table 7 shows the test results.

Table 7. Results of API Testing

Test No	Result (ms)	Test No	Result (ms)
1	5000	26	2377
2	4690	27	2393
3	4680	28	2351
4	4565	29	2386
5	4502	30	2367
6	4310	31	2350
7	4011	32	2363
8	3677	33	2392
9	2804	34	2398
10	2388	35	2385
11	2380	36	2395
12	2395	37	2379
13	2383	38	2357
14	2373	39	2378
15	2362	40	2360
16	2384	41	2370
17	2353	42	2356
18	2391	43	2374
19	2352	44	2397
20	2369	45	2354
21	2371	46	3668
22	2390	47	3980
23	2358	48	4987
24	2399	49	5268
25	2359	50	5691
Average Acceptance		**2946**	

In the test table, the system access time to the API has decreased in experiments 1 to 10. Then in experiment 11 onwards, the data processing time is stable until it approaches the 49th experiment, the speed drops to nearly 6000 ms.

4. CONCLUSION

This study aims to complete the integration between hotel booking transactions, transportation bookings, and tour ticket bookings, which previously we had to do one by one. Therefore they become integrated so that transactions can be made simultaneously as one transaction. We have presented an approach to integrating hotel booking, transportation booking, and tour ticket booking using web service technology with REST architecture. Web service provides services that can be used by clients based on their needs. It offers an opportunity for the use of web services as a tool for integration. By using RESTful web service technology, web services can be accessed easily by clients with different platforms to perform certain operations. The solution to this problem is implemented in the user software using Rest-API technology. The use of Rest-API technology provides convenience in terms of scalability and reusability in software development.

We tested the validity of the built API. Testing is done by testing API calls. The API call testing conducted found that the average time was 2946 ms from 50 trial calls by ten different users accessing the system simultaneously.

This study created a web service that integrates hotel booking, transportation booking, and tour ticket booking. This study answers the gap from previous research. We hope that this study contributes to a more in-depth analysis of the REST web service's scalability attributes. For further investigation, we recommend adding API connections with the providers of travel location review information so that more details will be available in one particular place.

REFERENCES

Anugrah, C. S., Santoso, H. B., and Budi, I., Android-Based Halal Tourism Application Design Using User Centered Design Method, *Seminar Nasional Aptikom (Semnastik) 2019*, 314–321, 2019.

Barbaglia, G., Murzilli, S., and Cudini, S., Definition of REST web services with JSON schema, *Software: Practice and Experience*, 907–920, 2016.

Chemutai, N., *Mobile application for long-distance vehicles booking of passengers in Kenya*, 2019.

Chen, F., Lu, C., Wu, H., and Li, M., A semantic similarity measure integrating multiple conceptual relationships for web service discovery, *Expert Systems with Applications*, *67*, 19–31, 2017.

Garcia, L. M., Aciar, S., Mendoza, R., and Puello, J. J., Smart tourism platform based on microservice architecture and recommender services, In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Vol. 10995 LNCS*, Springer International Publishing, 2018.

Hu, W., Huang, Y., Liu, X., and Xu, C., Study on REST API Test Model Supporting Web Service Integration, *Proceedings - 3rd IEEE International Conference on Big Data Security on Cloud, BigDataSecurity 2017, 3rd IEEE International Conference on High Performance and Smart Computing, HPSC 2017 and 2nd IEEE International Conference on Intelligent Data and Securit*, 133–138, 2017.

Jnr, B. A., Applying Enterprise Architecture for Digital Transformation of Electro Mobility towards Sustainable Transportation, *SIGMIS-CPR'20: Proceedings of the 2020 on Computers and People Research Conference*, 38–46, 2020.

Listyani, H. T., Handojo, A., and Novianus, H., *Online Payment Transaction Simulation with Petra Christian University Canteen Case Study*. *6*(1), 56–62, 2016.

Liu, Jian, Yang, M., Zhang, L., and Zhou, W., An effective biomedical data migration tool from resource description framework to JSON, *Database*, *2019*(1), 1–9, 2019.

Liu, Jianxiao, Tian, Z., Liu, P., Jiang, J., and Li, Z., An approach of semantic web service classification based on naive bayes, *Proceedings - 2016 IEEE International Conference on Services Computing, SCC 2016*, 356–362, 2016.

Nesi, P., Badii, C., Bellini, P., Cenni, D., Martelli, G., and Paolucci, M., Km4City Smart City API: An Integrated Support for Mobility Services, *2016 IEEE International Conference on Smart Computing, SMARTCOMP 2016*, 2016.

Neumann, A., Laranjeiro, N., and Bernardino, J., An Analysis of Public REST Web Service APIs, *IEEE Transactions on Services Computing*, 1–14, 2018.

Paik, H. Y., Lemos, A. L., Barukh, M. C., Benatallah, B., and Natarajan, A., Web service implementation and composition techniques, In *Web Service Implementation and Composition Techniques*, 2017.

Parra, F., and María, L., *Chatbot API: A service to develop text-based interfaces*, 2017.

Perwira, R., and Santosa, B., Web Service Implementation on Academic Data Integration with Dikti Database Replicas, *Telematika*, *14*(01), 1–11, 2017.

Putra, M. G. L., and Putera, M. I. A., Comparative Analysis of Soap and Rest Methods Used in the Flask Framework to Build Web Services, *SCAN - Jurnal Teknologi Informasi Dan Komunikasi*, *14*(2), 1–7, 2019.

Rachmaniah, M., Ardiansyah, H. I., and Rachmansyah, I., Web-based Marketplace to Support Ecotourism e-Commerce, *IOP Conference Series: Earth and Environmental Science*, *187*(1), 2018.

Rafalimanana, H. F., Razafindramintsa, J. L., Ratovondrahona, A. J., Mahatody, T., and Manantsoa, V., Publish a Jason Agent BDI Capacity as Web Service REST and SOAP, *Smart Innovation, Systems and Technologies*, *146*, 163–171, 2020.

Rauf, I., Vistbakka, I., and Troubitsyna, E., Formal Verification of Stateful Services with REST APIs Using Event-B, *Proceedings - 2018 IEEE International Conference on Web Services, ICWS 2018 - Part of the 2018 IEEE World Congress on Services*, 131–138, 2018.

Sohaib, O., and Kang, K., E-commerce web accessibility for people with disabilities, *Lecture Notes in Information Systems and Organisation*, *22*, 87–100, 2017.

BIOGRAPHIES

Faiza Renaldi is a lecturer in informatics, Faculty of Science and Informatics, Universitas Jenderal Achmad Yani Indonesia. He received his master of business informatics at Universiteit Utrecht, The Netherlands, in 2006. His research interests are health informatics, information systems/information technology management, e-government, agile project management, and IT entrepreneurship.

Hary Suryanto is a final year undergraduate student in informatics, Universitas Jenderal Achmad Yani, Indonesia. His primary interests are mobile app development and eGovernment.

Agya Java Maulidin is a research assistant at informatics, Universitas Jenderal Achmad Yani, Indonesia. His primary research interests are web services technologies, NoSQL technology, and mobile-based application.

Irma Santikarama received her Bachelor's degree in Information System from Universitas Kristen Maranatha and Master's degree in Informatics from Institut Teknologi Bandung. Now, she is a lecturer in the Informatics Department, Faculty of Science and Informatics, Universitas Jenderal Achmad Yani. Her research area includes the information architecture, information management, and business process redesign.

Abdul Talib Bon is a professor of Production and Operations Management in the Faculty of Technology Management and Business at the Universiti Tun Hussein Onn Malaysia since 1999. He has a Ph.D. in Computer Science, which he obtained from the Universite de La Rochelle, France, in 2008. His doctoral thesis was on the topic Process Quality Improvement on Beltline Moulding Manufacturing. He studied Business Administration in the Universiti Kebangsaan Malaysia, for which he was awarded the MBA in the year 1998. He has a bachelor's degree and a diploma in Mechanical Engineering, which he obtained from the Universiti Teknologi Malaysia. He received his postgraduate certificate in Mechatronics and Robotics from Carlisle, United Kingdom, in 1997. He had published more than 150 International Proceedings and International Journals and eight books. He is a member of MSORSM, IIF, IEOM, IIE, INFORMS, TAM, and MIM.

