IMPLEMENTASI WEB SERVICE RESTFUL API PADA APLIKASI SHAMOSTORE BERBASIS ANDROID MENGGUNAKAN FLUTTER DAN LARAVEL SANCTUM

Nur Fauzi[1*], Mardi Hardjianto[2]

[1,2]Fakultas Teknologi Informasi, Teknik Informatika, Universitas Budi Luhur, Jakarta, Indonesia

Email: [1*]1711501948@studentbudiluhur.ac.id, [2]mardi.hardjianto@budiluhur.ac.id
(* : corresponding author)

Abstrak-Shamostore hanya mempunyai *web* saja untuk kegiatan berbisnis dan belum memasuki ranah mobile seperti *Android* dan belum memiliki API serta autentikasi pengguna. Dengan adanya masalah tersebut maka solusi terbaik yang dapat membantu pemilik dari Shamostore agar dapat memudahkan dalam berbisnis terutama di bidang *fashion* adalah membuat aplikasi berbasis *Android* menggunakan Flutter dan RESTful API serta Laravel Sanctum untuk menangani proses autentikasi atau otorisasi pengguna. Tujuan penelitian ini Mengimplementasikan Laravel sanctum untuk autentikasi token pada *Web Service* RESTful API pada saat login maupun register. Dengan menggunakan metode yang dilakukan oleh Gilvy langgawan yaitu pendekatan *RESTful API*, peneliti membuat layanan API yang dapat digunakan oleh aplikasi lain dan layar untuk aplikasi mobile dan klien *web*. Hasil dari penelitian sebagai bagian dari penelitian ini Laravel Sanctum digunakan sebagai factor tambahan untuk melakukan proses autentikasi atau otorisasi dalam aplikasi serta menghasilkan token yang dimana autentikasi atau otorisasinya ditangani melalui *cookie*.

Kata Kunci: web service, RESTful API, API, flutter, laravel, laravel sanctum.

IMPLEMENTATION OF WEB SERVICE RESTFUL API ON ANDROID-BASED SHAMOSTORE APPLICATION USING FLUTTER AND LARAVEL SANCTUM

Abstract-*Shamostore only has a web for business activities and has not entered the mobile realm like Android and does not yet have an API and user authentication. With these problems, the best solution that can help Shamostore owners make it easier to do business, especially in the fashion sector is to create Android-based applications using Flutter and RESTful APIs and Laravel Sanctum to handle the user authentication or authorization process. The purpose of this study is to implement Laravel sanctum for token authentication on the RESTful API Web Service at login and register. Using the method used by Gilvy Langgawan, namely the RESTful API approach, researchers create API services that can be used by other applications and screens for mobile applications and web clients. The results of the research as part of this research Laravel Sanctum is used as an additional factor to carry out the authentication or authorization process in the application and generate tokens where authentication or authorization is handled via cookies.*

Keywords*: web service, RESTful API, API, flutter, laravel, laravel sanctum.*

1. PENDAHULUAN

Perkembangan teknologi saat ini memberi pengaruh besar pada individu, organisasi maupun instansi pemerintahan [1]. Teknologi berkembang bertujuan untuk membantu meringankan pekerjaan manusia. Oleh karena itu banyak dikembangkan berbagai aplikasi [2] dengan berbagai bahasa program yang dengan mudah diunduh dengan *smartphone* sesuai dengan kebutuhan pengguna [3]. Salah satu jenis aplikasi yang banyak digunakan yaitu yang memberi informasi kepada penggunanya [4].

ShamoStore merupakan toko *fashion* yang menjual berbagai macam baju dan sepatu bermerk seperti Nike, Trasher, Adidas dan lain-lain. Pada musim pandemi Covid-19 ini pendapatan toko Shamostore mengalami penurunan dan para pelanggan tidak berani datang dan melakukan kegiatan belanja dan transaksi secara tatap muka. Agar dapat memudahkan toko Shamostore melakukan penjualan, oleh karena itu dibuatlah aplikasi *web service* berbasis RESTful API dan *Android* untuk Shamostore agar pelanggan dengan mudah melakukan

pembelian dan bertransaksi secara *online* dirumah tanpa takut terpapar virus Covid-19. RESTful merupakan arsitektur yang umum digunakan dalam desain layanan online. Menggunakan HTTP (*Hyper Text Mechanism*) sebagai protokol transmisi data [5].

RESTful merupakan arsitektur yang umum digunakan dalam desain layanan online. Menggunakan *HTTP (Hyper Text Mechanism*) sebagai protokol transmisi data. Keuntungan menggunakan RESTful API yaitu[6] :

a. Independensi bahasa dan platform
b. Lebih sederhana untuk dibuat daripada SOAP
c. Mudah dipahami dan tidak memerlukan alat apa pun
d. Ringkas, tidak memerlukan lapisan komunikasi tambahan (pesan)
e. Secara desain dan konsep, ini lebih mirip dengan web.

Tujuan dari penelitian ini yaitu "Mengimplementasikan Laravel sanctum untuk autentikasi token pada *Web Service* RESTful API pada saat login maupun register".

Dengan menggunakan metode yang dilakukan oleh Gilvy langgawan yaitu pendekatan *RESTful API*, peneliti membuat layanan API yang dapat digunakan oleh aplikasi lain, dan layar untuk aplikasi mobile dan klien *web*, sebagai bagian dari penelitian ini. Layanan API yang tenang memerlukan akses data dan standar penyimpanan di atas standar minimum.

SOAP adalah protokol yang berisi sekumpulan aturan yang mengatur komunikasi antar komputer. Ide dasarnya adalah dua aplikasi, dengan tidak memperdulikan operating system, bahasa pemrograman, atau detil implementasi teknis yang lain, dapat menggunakan informasi secara bersama-sama dengan menggunakan pesan yang dikodekan dalam suatu cara dimana kedua aplikasi tersebut dapat mengerti. Pesan tersebut berupa data XML. Disini, SOAP berperan menyediakan standar cara untuk menyusun pesan XML tersebut. SOAP adalah XML, dengan kata lain, SOAP adalah aplikasi dari spesifikasi XML, dapat menggunakan standard XML seperti XML Schema dan XML Namespaces untuk pendefinisian dan fungsi/penggunaan. Secara konseptual SOAP dapat dianggap sebagai DCOM versi XML [7]. SOAP merupakan mekanisme lain yang memungkinkan penggunaan remote procedure call. SOAP bersifat netral platform, netral bahasa dan tidak bergantung pada suatu objek model. Sehingga SOAP-*enabled distributed application* dapat menjangkau beragam operating sistem, dimana terdiri dari objek yang berasal dari vendor yang berbeda, ditulis pada bahasa yang berbeda, dan didasarkan pada objek model yang berbeda.

API menerima permintaan dalam bentuk JSON dari aplikasi, dan server web kemudian melakukan kueri untuk mengakses *database*, seperti yang ditunjukkan pada diagram di atas [8]. Setelah selesai, *web server* akan memberikan data atau hasil permintaan respon ke aplikasi [9]. Maka dari itu *Web Service* berbasis RESTFul API adalah salah satu cara yang cocok untuk diterapkan di Shamo Store.

2. METODE PENELITIAN

2.1 Data Penelitian

Pada penelitian yang dilakukan, data yang dipakai saat pengujian yaitu data produk dari Shamostore. Data produk tersebut berisi informasi produk seperti foto produk, nama produk, kategori, harga dan stok nantinya akan di input melalui web Admin dan di publish di aplikasi mobile Shamostore. Berikut ini data produk yang telah dibuat.

SHAMOSTORE

Alamat : Jl. Melawai , Kec. Kebayoran. Baru, Kota Jakarta Selatan

Telepone: 082147371873

Instagram : @shamotore.id Email :Shamostore@gmail.com

LAPORAN DAFTAR DATA PRODUK SEPATU

Tanggal : 4-May-2022

NO	FOTO PRODUK	NAMA PRODUK	KATEGORI	STOK	HARGA
1		Ultra 4D 5 Shoes	Running	4	Rp.2.000.000
2		Ultraboots 20 Shoes	Running	4	Rp.3.500.000
3		LEGO® SPORT SHOES	Training	4	Rp.4.000.000
4		Fortarun Running Shoes 2020	Training	4	Rp.2.000.000

Gambar 1. Data Penelitian Produk Shamostore

Selain data produk, ada data omset dari toko Shamostore, berikut data omzet yang telah diberikan.

Gambar 2. Data Omzet Toko Shamostre

Pada saat pandemi Covid-19 menyebabkan omset penjualan toko sepatu menurun secara signifikan tiap tahunnya, bisa dilihat di tahun 2019 omzet senilai 40 juta *Menu*run menjadi 30 juta di tahun 2020. Dan di tahun 2021 menjadi 15 juta. Dan secara grafik di tahun 2022 terjadi kenaikan omzet penjualan sepatu dari yang 15 juta di tahun 2021 menjadi 20 juta di tahun 2022. Dengan pembuatan aplikasi *web* dan mobile ini diharapkan dapat meningkatkan omset dari toko Shamostore.

2.2 Metode Pembanding

Pada metode yang dilakukan oleh Gilvy langgawan berikut metode pembanding dari REST dan SOAP. SOAP adalah protokol yang berisi sekumpulan aturan yang mengatur komunikasi antar komputer. Ide dasarnya adalah dua aplikasi, dengan tidak memperdulikan operating system, bahasa pemrograman, atau detil implementasi teknis yang lain, dapat menggunakan informasi secara bersama-sama dengan menggunakan pesan yang dikodekan dalam suatu cara dimana kedua aplikasi tersebut dapat mengerti. Pesan tersebut berupa data XML. Disini, SOAP berperan menyediakan standar cara untuk menyusun pesan XML tersebut. SOAP adalah XML, dengan kata lain, SOAP adalah aplikasi dari spesifikasi XML, dapat menggunakan standard XML seperti XML Schema dan XML Namespaces untuk pendefinisian dan fungsi/penggunaan. Secara konseptual SOAP dapat dianggap sebagai DCOM versi XML [7]. SOAP merupakan mekanisme lain yang memungkinkan penggunaan remote procedure call. SOAP bersifat netral platform, netral bahasa dan tidak bergantung pada suatu objek model. Sehingga SOAP-*enabled distributed application* dapat menjangkau beragam operating sistem, dimana terdiri dari objek yang berasal dari vendor yang berbeda, ditulis pada bahasa yang berbeda, dan didasarkan pada objek model yang berbeda.

Table 1. Rancangan Pengujian RESTful API

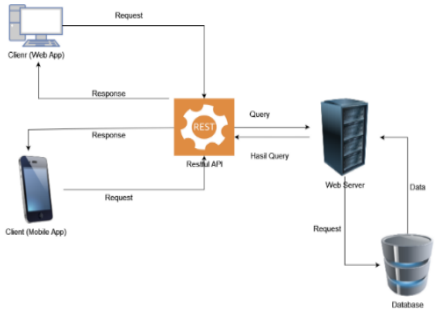
SN	SOAP REQUEST AND RESPON	REST REQUEST AND RESPON
1	2409	121
2	2409	146
3	2409	233
4	2409	118
5	2409	120
6	2409	26
7	2409	438

Dari data yang diperoleh yang ada pada Tabel 3.1 dan bahwa untuk web service yang SOAP dan REST api dari tabel tersebut, REST memiliki performance yang lebih bagus dan waktu yang lebih cepat dibandingkan dengan SOAP untuk pengetesan request dan respon untuk web service [10].

2.3 Penerapan Metode Penelitian

Dengan menggunakan pendekatan *RESTful API*, penulis membuat layanan API yang dapat digunakan oleh aplikasi lain, dan layar untuk aplikasi mobile dan klien *web*, sebagai bagian dari penelitian ini. Layanan API yang tenang memerlukan akses data dan standar penyimpanan di atas standar minimum.

API menerima permintaan dalam bentuk JSON dari aplikasi, dan server web kemudian melakukan kueri untuk mengakses *database*, seperti yang ditunjukkan pada diagram di atas [11]. Setelah selesai, *web server* akan memberikan data atau hasil permintaan respon ke aplikasi Anda.



Gambar 3. Metode Penelitian

2.4 Rancangan Pengujian

Dalam pengujian *RESTful API* ini dilakukan pengujian API yang telah dibuat dengan tujuan agar API ini sesuai dengan hasil rancangan yang telah dirancang serta menghasilkan satu kesimpulan yaitu apakah sistem tersebut sesuai dengan yang diharapkan.

Tabel 2. Rancangan Pengujian RESTful API

No	API	Request Method	Response Code	Status
1	Register	POST	200	Ok
2	Login	POST	200	Ok
3	Checkout	POST	200	Ok
4	Transaction	GET	200	Ok
5	Products	GET	200	Ok
6	Categories	GET	200	Ok
7	User	GET	200	Ok
8	Edit User	POST	200	Ok
9	Logout	GET	200	Ok

Agar server tahu apa yang diinginkan klien, ada prosedur untuk setiap permintaan:

a. Jika ingin menerima data dari situs web atau server, Anda harus menggunakan metode GET HTTP.
b. Teknik permintaan HTTP ini, yang memasukkan data ke dalam tubuh saat permintaan dikirim, dikenal sebagai POST.
c. Untuk memperbarui data sumber daya, HTTP PUT adalah cara permintaan HTTP yang lebih disukai Data dalam sumber daya dapat dihapus menggunakan teknik DELETE HTTP Request.

Untuk mengirimkan informasi yang diminta, kode respons HTTP digunakan. Kode respons HTTP RESTful API dijelaskan sebagai berikut [12]:

a. Dalam hal ini, itu berarti permintaan berhasil (200)
b. Permintaan yang berhasil ditunjukkan oleh kode respons 201 Diproduksi, yang menunjukkan bahwa data telah dibuat. Kode ini digunakan untuk memverifikasi bahwa permintaan PUT atau POST telah berhasil.
c. Permintaan Buruk: Kode respons ini digunakan untuk menunjukkan bahwa permintaan dibuat secara keliru atau bahwa data yang diberikan tidak ditemukan (400)
d. Akses ke sumber daya akan ditolak jika permintaan dikembalikan dengan kode status 401 Tidak Diotorisasi.
e. Jika Anda mendapatkan kode response 404 Tidak Terletak, itu berarti sumber daya yang Anda minta tidak dapat ditemukan.

3. HASIL DAN PEMBAHASAN

3.1 Lingkungan Percobaan

Standar yang digunakan untuk membangun aplikasi RESTful API ini juga harus mendukung aplikasi ini. Untuk mendukung aplikasi ini, persyaratan berikut harus dipenuhi:

3.1.1 Spesifikasi Hardware

Perangkat keras implementasi untuk aplikasi RESTful API ini sebagai berikut [13]:

a. Processor Intel® Core™ i5 5-6200U CPU @ 2.30GHz 2.40 GHz
b. RAM 8 GB
c. Laptop Asus X456UR
d. SSD 256 GB

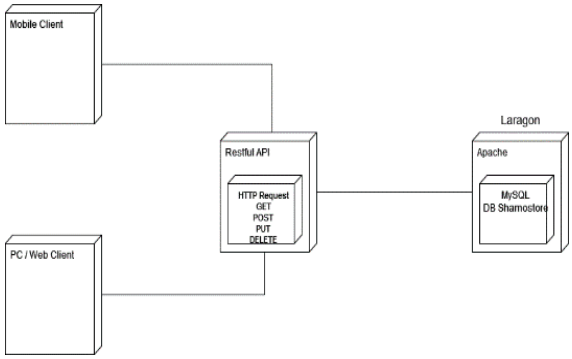
3.1.2 Spesifikasi Software

Ini adalah spesifikasi software yang dibutuhkan untuk membuat RESTful API [14]:

a. OS Microsoft Windows 10 64-bit
b. Bahasa Pemrograman PHP (Laravel) dan Dart (Flutter)
c. Database MySQL
d. Visual Studio Code
e. Emulator Android Studio (Android Q)
f. Laragon

3.1.3 *Deployment Diagram*

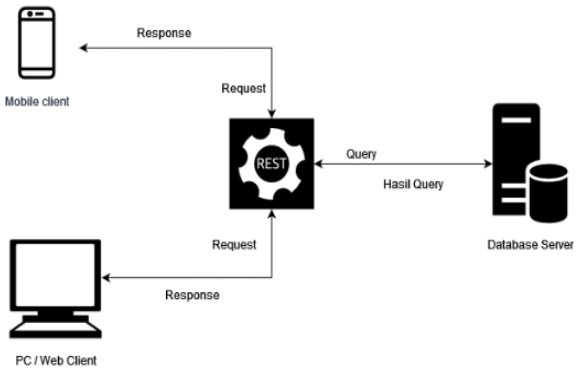
Menunjukan bahwa tata letak sebuah sistem secara fisik beserta Gambaran *hardware* dan *software* dari sistem yang dirancang. Berikut ini adalah Gambar *Deployment Diagram* implementasi *web service* menggunakan metode Rest API [15].



Gambar 4. *Deployment Diagram*

3.2 Implementasi Metode

Pada aplikasi Shamostore ini menggunakan metode Rest API. Berikut adalah implementasi dari Rest API:



Gambar 5. Implementasi Metode RESTful API

Aplikasi mengirimkan permintaan ke API dalam bentuk JSON, dan API merespons dengan mengeksekusi *query* pada database. Setelah selesai, server web akan memberikan data atau hasil permintaan respons ke aplikasi Anda.

3.3 Hasil Pengujian

Dari hasil pengujian yang dilakukan, diperoleh hasil dari pengujian *RESTful API* yang telah dirancang sebelumnya. Berikut hasil pengujian RESTful API untuk aplikasi Shamostore.

3.3.1 Hasil Pengujian Pada RESTful API Web

a. Hasil Pengujian RESTful API Register

Pada pengujian API Register, akan dikirimkan data user berupa Name, Username, Email, dan password. Kemudian API akan menghasilkan status code, status, dan message serta nantinya akan menghasilkan token secara otomatis yang bertipe Bearer. Hasil pengujian RESTful API Register dapat dilihat pada Gambar 6

```
{
    "meta": {
        "code": 200,
        "status": "success",
        "message": "User Registered"
    },
    "data": {
        "access_token": "18|
            OzSOBTi3aYiPYgQEgdBQA4VjwmspaEP24xiSK61t",
        "token_type": "Bearer",
        "user": {
            "id": 9,
            "name": "sarah wijayanto",
            "email": "sarahskuy@gmail.com",
            "username": "sarahskuyy",
            "roles": "USER",
            "email_verified_at": null,
            "current_team_id": null,
            "profile_photo_path": null,
            "created_at": "2022-07-31T15:36:44.000000Z",
```


Gambar 6. Hasil Pengujian RESTful API *Register*

b. Hasil Pengujian RESTful API Login

User yang sebelumnya telah menggunakan register API mengirimkan alamat email dan password untuk diuji menggunakan API Login. Token tipe Bearer secara otomatis dihasilkan sebagai bagian dari respons dari API Login. Hasil respon yang diperoleh sama dengan hasil pengujian registrasi API, yaitu kode status, status, dan pesan. Nomor kesalahan 500 dihasilkan saat pengguna mengirimkan alamat email yang tidak terdaftar atau kata sandi yang tidak valid ke API. Gambar 7 menunjukkan hasil pengujian RESTful API Login.

```
{
    "meta": {
        "code": 200,
        "status": "success",
        "message": "Authenticated"
    },
    "data": {
        "access_token": "19|
            pGiewItN3ftmJjlR3euSOSFLuCzxw2ni6Fekhbuq",
        "token_type": "Bearer",
        "user": {
            "id": 9,
            "name": "sarah wijayanto",
            "email": "sarahskuy@gmail.com",
            "username": "sarahskuyy",
            "roles": "USER",
            "email_verified_at": null,
            "current_team_id": null,
            "profile_photo_path": null,
            "created_at": "2022-07-31T15:36:44.000000Z",
```


Gambar 7. Hasil Pengujian RESTful API *Login*

c. Hasil Pengujian RESTful API Checkout

Pada pengujian API *Checkout*, *user* mengirimkan data token yang telah didapatkan pada API *Login* maupun *Register* dan data *Request body*. Kemudian respon API *checkout* menghasilkan data *transaction items* dan *transaction details* dari pengguna seperti status, message dan keluaran data yang dihasilkan seperti users_id, address, total_price, shipping_price, dan status. Jika token yang dikrimkan salah maka akan keluar pesan *unauthorized*. Hasil pengujian RESTful API *Checkout* dapat dilihat pada Gambar 8.

```
{
    "meta": {
        "code": 200,
        "status": "success",
        "message": "Transaksi Berhasil"
    },
    "data": {
        "users_id": 9,
        "address": "My Home no.123",
        "total_price": 2000,
        "shipping_price": 100,
        "status": "PENDING",
        "updated_at": "2022-07-31T15:39:29.000000Z",
        "created_at": "2022-07-31T15:39:29.000000Z",
        "id": 19,
        "items": [
            {
                "id": 44,
                "users_id": 9,
                "oroducts id": 1.
```


Gambar 8. Hasil Pengujian RESTful API *Checkout*

d. Hasil Pengujian RESTful API Transaction

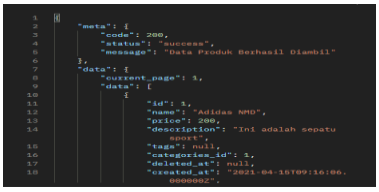
Pada pengujian API *Transaction*, *user* mengirimkan data token yang telah didapatkan pada API *Login* maupun *Register* dan data *Request body*. Kemudian respon API *Transaction* menghasilkan data *transaction items* dan *transaction details* dari pengguna seperti status, message dan keluaran data yang dihasilkan seperti users_id, address, total_price, shipping_price, dan status. Jika token yang dikrimkan salah maka akan keluar pesan *unauthorized*..

```
{
    "meta": {
        "code": 200,
        "status": "success",
        "message": "Data list transaksi berhasil
            diambil"
    },
    "data": {
        "current_page": 1,
        "data": [
            {
                "id": 19,
                "users_id": 9,
                "address": "My Home no.123",
                "total_price": 2000,
                "shipping_price": 100,
                "status": "PENDING",
                "payment": "MANUAL",
                "deleted_at": null,
                "created_at": "2022-07-31T15:39:29.
                    000000Z",
                "updated_at": "2022-07-31T15:39:29.
```


Gambar 9. Hasil Pengujian RESTful API *Transaction*

e. Hasil Pengujian RESTful API Products

Pada API *Products* respon API ini menghasilkan informasi produk seperti id, name, price, description, tags, categories_id, category. Hasil pengujian RESTful API *Products* dapat dilihar pada Gambar 10 berikut.



```
{
    "meta": {
        "code": 200,
        "status": "success",
        "message": "Data Produk Berhasil Diambil"
    },
    "data": {
        "current_page": 1,
        "data": [
            {
                "id": 1,
                "name": "Adidas NMD",
                "price": 200,
                "description": "Ini adalah sepatu
                    sport",
                "tags": null,
                "categories_id": 1,
                "deleted_at": null,
                "created_at": "2021-04-16T09:16:06.
                    000000Z",
```


Gambar 10. Hasil Pengujian RESTful API *Products*

f. Hasil Pengujian RESTful API *Categories*

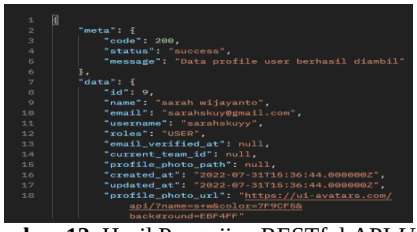
Pada API *Category* menghasilkan informasi data kategori berupa id, name, deleted_at, created_at, updated_at. Hasil pengujian RESTful API *Category* dapat dilihar pada Gambar 11 berikut

```
{
    "meta": {
        "code": 200,
        "status": "success",
        "message": "Data Kategori Berhasil Diambil"
    },
    "data": {
        "current_page": 1,
        "data": [
            {
                "id": 1,
                "name": "Sport",
                "deleted_at": null,
                "created_at": "2021-04-15T09:15:27.
                    000000Z",
                "updated_at": "2021-04-15T09:15:27.
                    000000Z"
            },
            {
                "id": 2,
```


Gambar 11. Hasil Pengujian RESTful API *Categories*

g. Hasil Pengujian RESTful API *User*

Pada pengujian API *User*, dikirimkan data token yang telah didapat dari API *Login* atau *register*, kemudian respons dari API *user* menghasilkan data *user* dari pengguna seperti id, name, email, username, dan roles. Hasil pengujian RESTful API *User* dapat diliha pada Gambar 12 berikut.

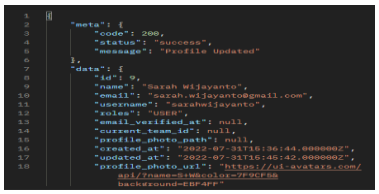


```
{
    "meta": {
        "code": 200,
        "status": "success",
        "message": "Data profile user berhasil diambil"
    },
    "data": {
        "id": 9,
        "name": "sarah wijayanto",
        "email": "sarahskuy@gmail.com",
        "username": "sarahskuyy",
        "roles": "USER",
        "email_verified_at": null,
        "current_team_id": null,
        "profile_photo_path": null,
        "created_at": "2022-07-31T15:36:44.000000Z",
        "updated_at": "2022-07-31T15:36:44.000000Z",
        "profile_photo_url": "https://ui-avatars.com/
            api/?name=s+w&color=7F9CF5&
            background=EBF4FF"
```


Gambar 12. Hasil Pengujian RESTful API *User*

h. Hasil Pengujian RESTful API *Edit User*

Pada pengujian API *Edit User*, dikirimkan data token yang telah didapat dari API *Login* atau *register* dan mengisi Request body berupa name, username, dan email, Hasil yang diterima dari API *Edit User* adalah perubahan informasi pada data *user*. Hasil pengujian RESTful API Edit *User* dapat dilihat pada Gambar 13 berikut.



```
{
    "meta": {
        "code": 200,
        "status": "success",
        "message": "Profile Updated"
    },
    "data": {
        "id": 9,
        "name": "Sarah Wijayanto",
        "email": "sarah.wijayanto@gmail.com",
        "username": "sarahwijayanto",
        "roles": "USER",
        "email_verified_at": null,
        "current_team_id": null,
        "profile_photo_path": null,
        "created_at": "2022-07-31T15:36:44.000000Z",
        "updated_at": "2022-07-31T15:45:42.000000Z",
        "profile_photo_url": "https://ui-avatars.com/
            api/?name=S+W&color=7F9CF5&
            background=EBF4FF"
```


Gambar 13. Hasil Pengujian RESTful API *Edit User*

3.3.2 Pengujian RESTful API Aplikasi Mobile

Pada pengujian RESTful API ini dilakukan pada aplikasi mobile Shamostore, berikut hasil pengujian RESTful API pada aplikasi mobile Shamostore.

a. Pengujian RESTful API Sign Up Aplikasi Mobile

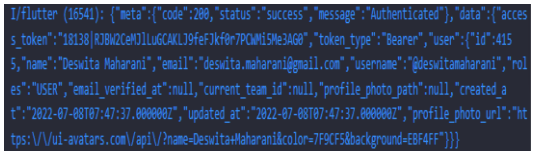
Pada pengujian Rest API Sign Up Aplikasi Mobile akan dikirimkan data user berupa Name, Username, Email, dan password. Kemudian API menghasilkan status code, status, dan message serta nantinya akan menghasilkan token secara otomatis yang bertipe Bearer.

```
I/flutter (16541): {"meta":{"code":200,"status":"success","message":"User Registered"},"data":{"acc
ess_token":"18137|E9L6IkxxwYQHeJWWyRNqmQGcXqLhqKYxdykQD96z","token_type":"Bearer","user":{"id":415
5,"name":"Deswita Maharani","email":"deswita.maharani@gmail.com","username":"@deswitamaharani","rol
es":"USER","email_verified_at":null,"current_team_id":null,"profile_photo_path":null,"created_a
t":"2022-07-08T07:47:37.000000Z","updated_at":"2022-07-08T07:47:37.000000Z","profile_photo_url":"ht
tps:\/\/ui-avatars.com\/api\/?name=Deswita+Maharani&color=7F9CF5&background=EBF4FF"}}}
```


Gambar 14. Pengujian RESTful API Sign Up Aplikasi Mobile

b. Pengujian RESTful API Sign In Aplikasi Mobile

Pada pengujian API Sign in Aplikasi Mobile, User mengirimkan data email dan password yang sebelumnya didaftarkan menggunakan API register, Hasil response yang didapat sama seperti hasil pengujian API register yaitu menghasilkan status code, status, dan message serta nantinya akan menghasilkan token secara otomatis yang bertipe Bearer. Ketika user mengirimkan data email yang tidak terdaftar atau password yang salah maka API akan merespon dengan menghasilkan status code 500 dan message authenticated.



```
I/flutter (16541): {"meta":{"code":200,"status":"success","message":"Authenticated"},"data":{"acces
s_token":"18138|RJBW2CeMJlLuGCAKLJ9feFJkf0r7PCWNi5Me3AG0","token_type":"Bearer","user":{"id":415
5,"name":"Deswita Maharani","email":"deswita.maharani@gmail.com","username":"@deswitamaharani","rol
es":"USER","email_verified_at":null,"current_team_id":null,"profile_photo_path":null,"created_a
t":"2022-07-08T07:47:37.000000Z","updated_at":"2022-07-08T07:47:37.000000Z","profile_photo_url":"ht
tps:\/\/ui-avatars.com\/api\/?name=Deswita+Maharani&color=7F9CF5&background=EBF4FF"}}}
```


Gambar 15. Pengujian RESTful API Sign In Aplikasi Mobile

c. Pengujian RESTful API Product Aplikasi Mobile

Pada API Products Aplikasi Mobile respon API ini diterima dengan menghasilkan informasi produk seperti id, name, price, description, tags, categories_id, category.

Gambar 16. Pengujian RESTful API Product Aplikasi Mobile

d. Pengujian RESTful API Checkout Aplikasi Mobile

Pada pengujian API Checkout Aplikasi Mobile, ketika user sedang login dan ingin melakukan transaksi setelah memilih barang yang diinginkan lalu melakukan checkout pada barang yang dipilih. Respon API Checkout pada aplikasi mobile menghasilkan data transaction items dan transaction details dari pengguna seperti status, message dan keluaran data yang dihasilkan seperti users_id, address, total_price, shipping_price, dan status.

Gambar 17. Pengujian RESTful API Product Aplikasi Mobile

4. KESIMPULAN

Berdasarkan pengujian dan implementasi yang dilakukan pada bab sebelumnya, maka dapat disimpulkan bahwa:

a. Dengan menggunakan metode yang dilakukan oleh Gilvy langgawan dengan pendekatan RESTful API membuat API sesuai dengan metode HTTP sehingga rute API menjadi lebih terstruktur, mudah dibaca, dan tidak memakan banyak kata.
b. 9 API dapat berfungsi dengan lancar di aplikasi Android Shamostore yang dibangun dengan Flutter dan Laravel Sanctum.
c. Keempat metode request HTTP dapat mempersingkat semua URL dengan URL yang sama dan hanya metode yang berbeda, dan waktu yang dibutuhkan untuk menjalankan API cukup cepat.

DAFTAR PUSTAKA

[1] D. setiawan, “Dampak Perkembangan Teknologi Informasi Dan Komunikasi Terhadap Budaya,” *J. Simbolika Res. Learn. Commun. Study*, vol. 4, no. 1, p. 62, 2018, doi: 10.31289/simbollika.v4i1.1474.
[2] Y. Soraya and J. Husna, “Motivasi Relawan Melalui Media Sosial Facebook Pada Gerakan Donasi Motor Pustaka Di Desa Pematang Pasir Kecamatan Ketapang Kabupaten Lampung Selatan,” *J. Ilmu Perpust.*, vol. 8, no. 2, pp. 256–266, 2020, [online]. available: https://ejournal3.undip.ac.id/index.php/jip/article/view/26814.
[3] D. Susanto and R. Intan, “Pembuatan Aplikasi Virtual Maket Rumah Berbasis Android,” 2017.
[4] R. Aditya, V. H. Pranata wijaya, and p. b. a. a. putra, “Rancang Bangun Aplikasi Monitoring Kegiatan Menggunakan Metode Prototype,” *J. Inf. Technol. Comput. Sci.*, vol. 1, no. 1, pp. 47–57, 2021, [online]. available: https://e-journal.upr.ac.id/index.php/jcoms/article/view/2955.
[5] N. Ramdhani and R. H. Al-Fadillah, “Web Service Dan Contoh Pengaplikasiannya,” no. march, 2020.
[6] R Rizal and A. Rahmatulloh, “Restful Web Service Untuk Integrasi Sistem Akademik Dan Perpustakaan Universitas Perjuangan,” *J. Ilm. Inform.*, vol. 7, no. 01, p. 54, 2019, doi: 10.33884/jif.v7i01.1004.
[7] M. M. Amin, “Interoperabilitas Perangkat Lunak Menggunakan Restful Web Service,” *Regist. J. Ilm. Teknol. Sist. Inf.*, vol. 4, no. 1, pp. 14–22, 2018, doi: 10.26594/register.v4i1.1129.
[8] A. Ramadhan, “Jurnal Teknologi Dan Manajemen Informatika Penerapan Aplikasi Android E-Payment Dan Pemesanan Layanan Pujasera,” vol. 8, no. 1, pp. 46–55, 2022.
[9] M. Arman, “Metode Pertahanan Web Server Terhadap Distributed Slow Http Dos Attack,” *Jatisi (Jurnal Tek. Inform. Dan Sist. Informasi)*, vol. 7, no. 1, pp. 56–70, 2020, doi: 10.35957/jatisi.v7i1.284.
[10] M. G. l. Putra and M. I. A. Putera, “Analisis Perbandingan Metode Soap Dan Rest Yang Digunakan Pada Framework Flask Untuk Membangun Web Service,” *Scan - J. Teknol. Inf. dan Komun.*, vol. 14, no. 2, pp. 1–7, 2019, doi: 10.33005/scan.v14i2.1480.
[11] Yaddarabullah and d. Lestari, “Perancangan Sistem Komunikasi Data Alat Pencatatan Meter,” *Infotekjar (Jurnal Nas. Inform. Dan Teknol. Jaringan)*, vol. 3, no. 1, pp. 49–54, 2018.
[12] A. Firdaus, S. widodo, A. Sutrisman, S. G. Fadhilah Nasution, and R. Mardiana, “Rancang Bangun Sistem Informasi Perpustakaan Menggunakan Web Sevice Pada Jurusan Teknik Komputer Polsri,” *J. Inform.*, vol. 5, no. 2, pp. 81–87, 2019.
[13] R. Choirudin and A. Adil, “Implementasi Rest Api Web Service Dalam Membangun Aplikasi Multiplatform Untuk Usaha Jasa,” *Matrik J. Manajemen, Tek. Inform. dan Rekayasa Komput.*, vol. 18, no. 2, pp. 284–293, 2019, doi: 10.30812/matrik.v18i2.407.
[14] W. Surya and A. Syahputra, “Perancangan Aplikasi Mobile E-Commerce Perangkat Elektronik Dengan Menggunakan Rest Api Berbasis Android,” *It (Informatic Tech. J.*, vol. 8, no. 2, p. 173, 2021, doi: 10.22303/it.8.2.2020.173-183.
[15] A. Imran and A. Rustianto, “Jurnal Informatika Terpadu,” J. *Inform. Terpadu*, vol. 7, no. 1, pp. 33–38, 2021, [online]. available: https://journal.nurulfikri.ac.id/index.php/jit.