Pembangunan *Website* Paket *Travel* dan Sewa Mobil Berbasis Arsitektur *Microservices*

Made Riksi Purnama Sadnya Agung[1], Thomas Adi Purnomo Sidhi[2], Patricia Ardanari[3]
Program Studi Informatika, Fakultas Teknologi Industri, Universitas Atma Jaya Yogyakarta
Jl. Babarsari No.43, Sleman 55281, Daerah Istimewa Yogyakarta, Indonesia
Email: [1]maderiksi20@gmail.com, [2]thomas.adi.ps@uajy.ac.id, [2]patricia.ardanari@uajy.ac.id

Abstract. *The tourism sector in Bali continues to grow, but Bali Travel Ride still faces challenges in managing bookings and finances due to manual systems. To address this, a microservices-based website was developed to integrate travel package and car rental booking services, simplifying business operations. This research includes literature study, system architecture design, and system development using Next.js on the front-end and Nest.js on the back-end, with payment integration through a Payment Gateway. The deployment process utilizes Docker, Nginx as a load balancer, and Cloudflare Tunnel for public access. The results of Black Box testing show that all features function according to specifications. The implementation of the microservices architecture has proven to enhance flexibility and scalability, allowing each service to be developed independently without affecting others, while simplifying the management of bookings, payments, and financial reports.*
Keywords: *website, booking, microservices*

Abstrak. *Sektor pariwisata Bali terus berkembang, namun Bali Travel Ride masih terkendala dalam pengelolaan pemesanan dan keuangan karena sistem manual. Untuk mengatasinya, dikembangkan website berbasis microservices yang mengintegrasikan layanan pemesanan paket travel dan sewa mobil agar mempermudah operasional bisnis. Penelitian ini mencakup studi literatur, perancangan arsitektur, serta pengembangan sistem menggunakan Next.js di front-end dan Nest.js di back-end dengan pembayaran menggunakan Payment Gateway. Proses deployment dilakukan dengan Docker, Nginx sebagai load balancer, dan Cloudflare Tunnel untuk akses publik. Hasil pengujian Black Box menunjukkan seluruh fitur berfungsi sesuai spesifikasi. Penerapan arsitektur microservices terbukti meningkatkan fleksibilitas dan scalability, memungkinkan setiap layanan dikembangkan secara independen tanpa mengganggu sistem lain, sekaligus mempermudah pengelolaan pemesanan, pembayaran, dan laporan keuangan.*
Kata Kunci: *website, pemesanan, microservices*

1. Pendahuluan

Berdasarkan data dari Badan Pusat Statistik (BPS) Provinsi Bali, pada April 2024 tercatat sebanyak 503.194 kunjungan wisatawan mancanegara ke Bali, meningkat 7,24% dibandingkan bulan sebelumnya yang berjumlah 469.227 kunjungan [1]. Kemudian pada Mei 2024 jumlah kunjungan naik lagi menjadi 544.601 atau meningkat 8,23% [2]. Data ini menunjukkan adanya tren positif terhadap peningkatan kunjungan wisatawan meskipun pertumbuhannya tidak selalu besar. Kondisi tersebut menjadi peluang yang signifikan bagi pelaku usaha di sektor pariwisata untuk mengembangkan layanan mereka, khususnya melalui penerapan teknologi digital agar operasional bisnis menjadi lebih baik dan mampu menjangkau pasar yang lebih luas.

Transformasi digital dalam sektor pariwisata sangat penting untuk meningkatkan efisiensi dan daya saing usaha. Menurut laporan tahunan *World Travel & Tourism Council (WTTC)*, yang mengukur dampak ekonomi dan ketenagakerjaan pariwisata di 185 negara dan 25 wilayah geografis atau ekonomi, sektor pariwisata berkontribusi sebesar 10,4% terhadap GDP global dengan nilai sekitar 8,9 triliun dolar AS pada tahun 2019. Digitalisasi telah membantu

pelaku usaha dalam mengoptimalkan operasional, memperluas pemasaran global, serta meningkatkan pendapatan [3]. Namun, penerapan digitalisasi masih menghadapi tantangan seperti rendahnya pemahaman terhadap teknologi dan infrastruktur digital yang belum memadai [4]. Oleh karena itu, pelaku usaha pariwisata di Bali perlu mengadopsi sistem digital yang terintegrasi untuk menjawab tantangan tersebut.

Bali Travel Ride merupakan usaha di bidang wisata dan transportasi yang menawarkan berbagai paket wisata serta layanan sewa mobil. Namun, proses pemesanan dan pengelolaan layanan masih dilakukan secara manual atau melalui aplikasi *chatting*, sehingga menyulitkan operasional bisnis. Untuk mengatasinya, dibutuhkan *website* dengan arsitektur *microservices* yang membantu proses pemesanan, pembayaran, *reschedule*, dan pembatalan pesanan, serta mendukung pencatatan pendapatan, pengeluaran, dan laba rugi. Sistem ini juga terintegrasi dengan *payment gateway* agar wisatawan dapat melakukan pembayaran dengan mudah dan aman. Arsitektur *microservices* dipilih karena mampu menangani sistem kompleks, memudahkan pengembangan dan *deployment* tiap layanan, serta meningkatkan *scalability* tanpa mengganggu layanan lainnya [5].

Berdasarkan rumusan masalah tersebut, tujuan dari penelitian ini adalah untuk merancang dan membangun *website* berbasis arsitektur *microservices* yang dapat menerima pesanan paket *travel* dan sewa mobil secara *online* dan dapat membantu operasional bisnis Bali Travel Ride. Mulai dari pengelolaan data paket *travel* dan sewa mobil, pengelolaan pesanan sampai dengan pencatatan keuangan.

2. Tinjauan Pustaka

Penelitian-penelitian sebelumnya terkait pengembangan *website* serupa menjadi acuan dalam penyempurnaan sistem yang dibangun. Evaluasi terhadap studi tersebut membantu menganalisis keunggulan dan keterbatasan sistem yang ada. Berdasarkan kajian tersebut, penelitian ini mengimplementasikan konsep dan pendekatan untuk merancang *website* layanan paket wisata dan sewa mobil dengan arsitektur *microservices*.

Penelitian pertama membahas arsitektur serupa yaitu jurnal berjudul *"*Implementasi Arsitektur *Microservice* Pada *back-end* Sistem Informasi Atlantas Berbasis *Website"*. Penelitian ini mengembangkan sistem informasi lalu lintas di Polres Metro Depok dengan arsitektur *microservices* menggunakan *framework* Lumen Laravel dan metode Rapid Application Development (RAD). Hasil pengujian Black Box menunjukkan tingkat keberhasilan 100%, dan pengujian integrasi mencapai 71%, menandakan sistem berjalan efektif dan terintegrasi [6].

Penelitian kedua membahas pengembangan *website* untuk biro *Smart Tour* di Bandar Lampung dengan judul "Pengembangan Sistem Informasi Web Jasa Tour dan Travel (Studi Kasus Smart Tour)". Aplikasi ini mengubah pemesanan tiket dan penyewaan mobil menggunakan PHP dan *framework* MVC secara digital. Pengujian Black Box menunjukkan bahwa sistem berhasil menggantikan proses manual menjadi pemesanan *online* yang tercatat dalam sistem [7].

Penelitian ketiga, yaitu berasal dari jurnal berjudul "Pengembangan Sistem Reservasi Rental Kendaraan dan Trip Wisata berbasis Web (Studi Kasus: G19 Tour & Travel)". Penelitian ini bertujuan untuk mengatasi masalah dari G19 Tour & Travel, yaitu pencatatan manual yang menghambat produktivitas. Penelitian ini mengembangkan sistem reservasi *online*. Sistem yang dikembangkan menggunakan *framework* Next.js di sisi klien dan Express.js di sisi server. Basis data yang digunakan adalah PostgreSQL. Dan pada bagian sistem pembayaran menggunakan Midtrans. Untuk pengujiannya, menggunakan teknik pengujian White Box dan Black Box. Hasil dari pengujian mendapat hasil yang sesuai dengan kebutuhan yang diharapkan [8].

Penelitian keempat, merupakan jurnal yang berjudul "Sistem Informasi Pemesanan Tiket Pada Indah Travel Berbasis Web". Penelitian ini bertujuan untuk membangun sistem informasi pemesanan tiket pada Indah Travel berbasis web. Sistem yang dibuat menggunakan bahasa pemrograman PHP dan basis data MySQL. Hasilnya, semua fungsi berjalan dengan baik seperti mengelola data, penyediaan daftar tiket, serta transaksi pembelian dan pembatalan pemesanan tiket sehingga semua fitur yang disediakan dapat memudahkan bagi pelanggan dan pengelola sistem [9].

Penelitian kelima membahas arsitektur serupa yaitu jurnal berjudul "Pengembangan Arsitektur *Microservice* pada *Learning Management System E-learning* Menggunakan Metode Web *Service*". Penelitian ini mengembangkan Learning Management System (LMS) berbasis *microservices* dengan beberapa layanan menggunakan Laravel dan Express.js, basis data MySQL, serta integrasi pembayaran Midtrans. Metode pengembangan menggunakan Software Development Life Cycle (SDLC) Waterfall dengan pengujian Black Box dan integrasi layanan. Hasil pengujian menunjukkan sistem berjalan baik dengan *response code* 200 saat semua layanan aktif dan 500 saat salah satu layanan tidak aktif, menandakan integrasi berjalan efektif [10].

Berdasarkan tinjauan pustaka yang sudah dipaparkan, ada beberapa hal yang akan diadopsi, arsitektur *microservice* pada sisi *back-end* untuk *scalability* tiap layanan. Sistem ini juga akan mengintegrasikan sistem pembayaran dengan *Payment Gateway*. Sistem ini menggunakan bahasa pemrograman Typescript untuk pengembangannya. Sistem ini mengimplementasikan arsitektur *microservices* pada sisi *back-end* dengan *framework* Nest.js. Kemudian, pada sisi *front-end* menggunakan *framework* Next.js. Dan untuk basis data akan menggunakan PostgreSQL dan Redis sebagai penyimpanan *caching*.

3. Metodologi Penelitian

Metode penelitian yang digunakan dalam pembangunan *website* paket *travel* dan sewa mobil berbasis *microservices* terdiri dari lima tahapan. Pertama, studi literatur dan perencanaan dilakukan untuk mengumpulkan referensi mengenai arsitektur *microservices*, *tools* yang digunakan, serta *payment gateway*. Kedua, desain sistem mencakup pembuatan Use Case Diagram (UCD), Entity Relationship Diagram (ERD), rancangan arsitektur *microservices*. Ketiga, implementasi kode pada *front-end* dan *back-end*.

Tahap keempat adalah pengujian untuk memastikan seluruh fungsi berjalan lancar. Pengujian dilakukan terhadap arsitektur *microservices* dan fungsionalitas sistem menggunakan metode Black Box *testing*, yaitu adalah metode pengujian yang mengevaluasi fungsionalitas perangkat lunak tanpa mempertimbangkan struktur internal atau implementasi kode. Pendekatan ini memungkinkan pengujian dilakukan dari sudut pandang pengguna untuk mendeteksi inkonsistensi terhadap spesifikasi kebutuhan tanpa memerlukan pengetahuan pemrograman, sehingga validasi sistem dapat dilakukan secara objektif dan menyeluruh [11].

Tahap terakhir adalah *deployment*, yaitu proses merilis sistem agar dapat digunakan secara publik. Setiap layanan dijalankan dalam bentuk *container* di server. Proses ini dikombinasikan dengan Nginx sebagai *load balancer* untuk mengatur lalu lintas jaringan. Selain itu, *tunneling* melalui Cloudflare digunakan agar layanan dapat diakses secara publik dengan aman dan stabil.

4. Hasil dan Diskusi

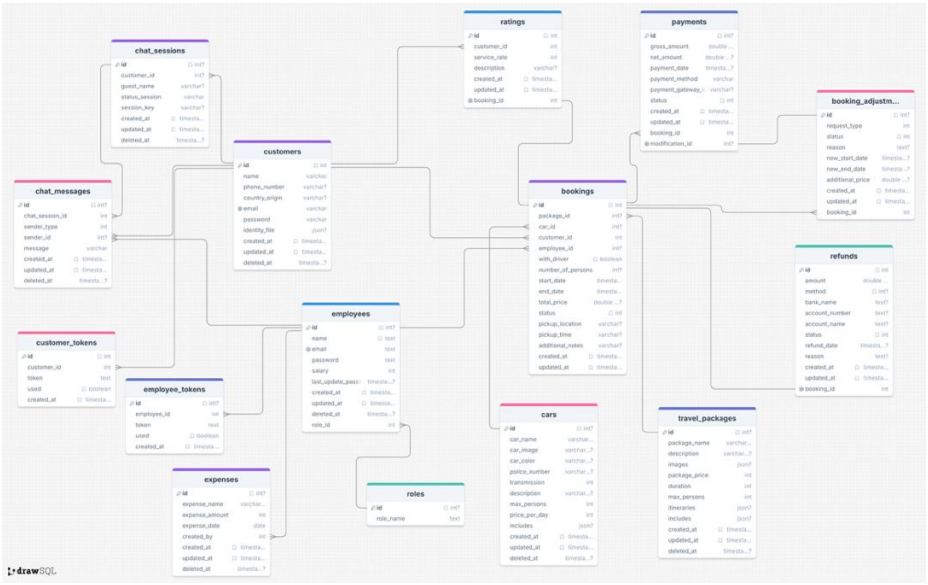
4.1 Fungsi Produk

Website paket *travel* dan sewa mobil ini dibangun untuk membantu operasional bisnis dari Bali Travel Ride. Pembangunan sistem *website* ini melibatkan tiga aktor. Seperti pada Gambar 1, dalam UCD tersebut terdapat aktor *owner*, admin, dan *customer*. Tiap aktor ini memiliki perannya masing-masing di dalam sistem.

Gambar 1. Diagram Use Case

4.2 Perancangan Data

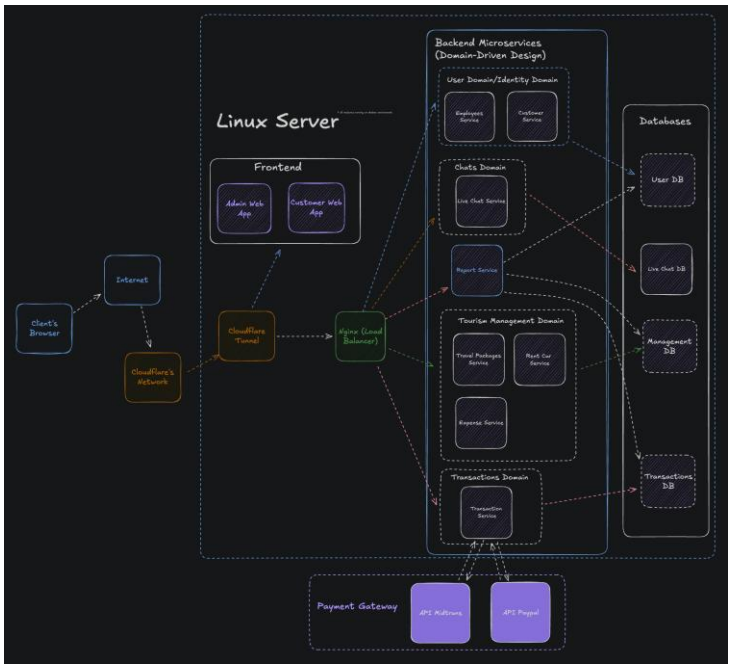
Dalam pembangunan *website* paket *travel* dan sewa mobil berbasis arsitektur *microservices* ini, dilakukan perancangan data untuk memvisualisasikan hubungan antar entitas di dalam sistem. Perancangan data ini menggunakan ERD. Pada Gambar 2, menampilkan entitas-entitas yang saling berhubungan dan lengkap dengan atribut yang dimiliki tiap entitas.



Gambar 2. Entity Relationship Diagram

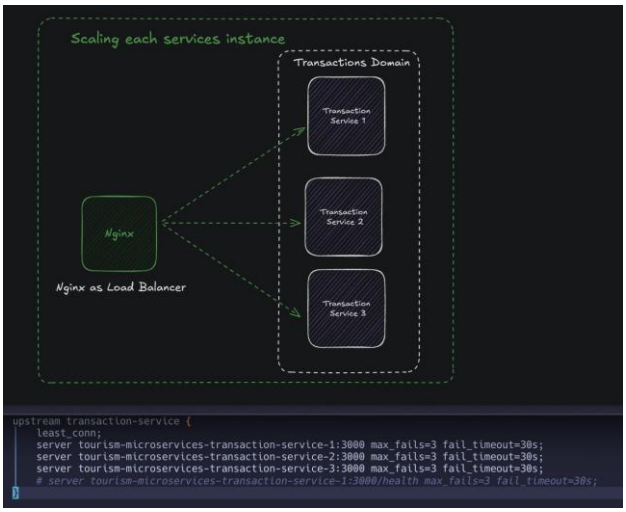
4.3 Perancangan Arsitektur

Pembangunan *website* paket *travel* dan sewa mobil ini menggunakan arsitektur *microservices* seperti ditunjukkan pada Gambar 3. Arsitektur ini membagi *back-end* menjadi beberapa layanan kecil sesuai fungsinya, dan *front-end* menjadi dua aplikasi, yaitu untuk *customer* dan untuk *admin/owner*. Cloudflare Tunnel digunakan sebagai *service discovery* dan publikasi layanan, sedangkan Nginx berperan sebagai *service discovery* di tingkat *back-end* sekaligus *load balancer*.



Gambar 3. Perancangan Arsitektur *Microservices*

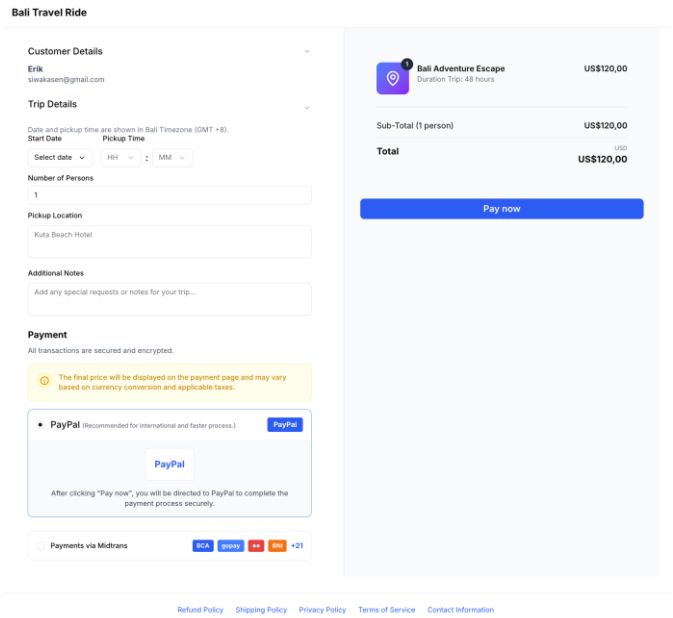
Pada Gambar 4 menunjukkan rancangan Nginx dalam melakukan *load balancing* pada layanan *transaction service*. Dan pada bagian bawah Gambar 4, menampilkan konfigurasi pada Nginx yang menggunakan algoritma *least connection* yang berarti mencari *service* dengan *traffic* paling rendah terlebih dahulu. Selain itu, tiap layanan dikonfigurasikan untuk mentoleransi *fail request* maksimal tiga kali dan *fail timeout* selama 30 detik.



Gambar 4. Perancangan *Load Balancing*

4.4 Implementasi Antarmuka

Pada Gambar 5 menampilkan antarmuka pemesanan paket *travel* untuk *customer*. Halaman ini menampilkan *Customer Details* yang berisi informasi *customer*, *Trip Details* untuk memilih tanggal keberangkatan, waktu penjemputan, jumlah peserta, serta lokasi penjemputan. *Customer* juga dapat menambahkan catatan tambahan terkait permintaan khusus pada kolom *Additional Notes*. Kemudian, bagian *Payment* menyediakan metode pembayaran melalui PayPal dan Midtrans. Setelah data diisi, *customer* dapat menekan tombol *Pay now* untuk melanjutkan ke proses pembayaran.



Gambar 5. Implementasi Antarmuka Pemesanan Paket Travel

Pada Kode 1 menunjukkan potongan kode fungsi `createBooking()` yang digunakan untuk membuat data pemesanan baru. Fungsi ini menerima data dari *customer* dan data pesanan. Sistem akan memvalidasi ketersediaan paket *travel* dengan fungsi `getPackageGrpc()`. Jika ditemukan, sistem menghitung total harga berdasarkan durasi perjalanan dan jumlah peserta, kemudian menyusun data pemesanan.

Kode 1. Implementasi Kode Pemesanan Paket Travel

```
public async createBooking(
    payload: BookingReqDto,
    customer: Customer,
    isRegister: boolean
  ): Promise<{data: {message: string;redirect_url: string;};}> {
    const queryRunner = this.dataSource.createQueryRunner();
    await queryRunner.connect();
    await queryRunner.startTransaction();
    try {
      let booking: Bookings;
      let product_name = '';
      let total_price = 0;
      let redirect_url = null;

      if (payload.package_id) {
        const packageData = await this.getPackageGrpc(payload);
        if (!packageData) {
          throw new HttpException('Package not found',HttpStatus.NOT_FOUND);
       }
        const startDate = new Date(payload.start_date);
```

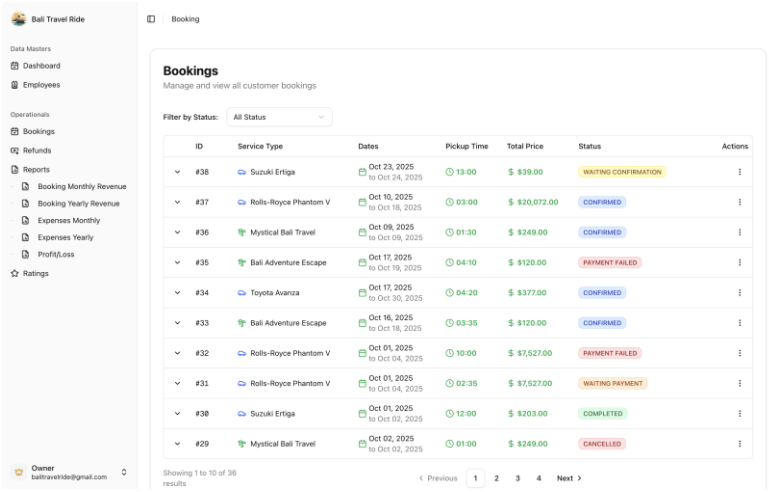
```
        const endDate = new Date(
          startDate.getTime() + packageData.duration * 1 * 60 * 60 * 1000
        );
        total_price = packageData.packagePrice * payload.number_of_persons;
        product_name = packageData.packageName;
        booking = await queryRunner.manager.save(Bookings, {
          package_id: payload.package_id,
          total_price: total_price,
          start_date: payload.start_date,
          end_date: endDate,
          customer_id: customer.id,
          number_of_persons: payload.number_of_persons,
          status: BookingStatus.WAITING_PAYMENT,
          pickup_location: payload.pickup_location,
          pickup_time: payload.pickup_time,
          additional_notes: payload.additional_notes,
        });
```

Pada Kode 2 menunjukkan potongan kode setelah kode pembuatan URL pembayaran setelah kode pembuatan pesanan. Pada potongan Kode 2 ini menampilkan pemisahan pembuatan URL pembayaran berdasarkan metode pembayaran yang di-*generate* menggunakan API *payment gateway*-nya masing-masing. URL ini digunakan di sisi *front-end* untuk menampilkan halaman pembayaran berdasarkan metode yang dipilih oleh *customer*.

Kode 2. Implementasi Kode Pembuatan URL Pembayaran Setelah Pemesanan

```
if (payload.payment_method === PaymentMethod.MIDTRANS) {
        const { redirect_url: midtrans_redirect_url } =
          await this.paymentService.createTransactionMidtrans(
            booking,
            product_name,
            total_price,
            customer,
            queryRunner
          );
        redirect_url = midtrans_redirect_url;
      } else if (payload.payment_method === PaymentMethod.PAYPAL) {
        const { redirect_url: paypal_redirect_url } =
          await this.paymentService.createOrderPaypal(
            booking,
            product_name,
            total_price,
            customer,
            queryRunner
          );
        redirect_url = paypal_redirect_url;
      }
      await queryRunner.commitTransaction();
      return {
        data: {
          message: 'Booking success',
          redirect_url: redirect_url,
        },
      };
```

Pada Gambar 6 menampilkan antarmuka data semua pesanan *customer* yang dikelola oleh admin. Halaman ini menampilkan daftar pesanan dengan informasi minimal. Terdapat juga fitur *Filter by Status* untuk pencarian pesanan berdasarkan status. Dan Setiap baris pada tabel dilengkapi kolom *Actions* untuk melakukan tindakan lebih lanjut terhadap pesanan yang dipilih.



Gambar 6. Implementasi Antarmuka Data Semua Pesanan

Pada Kode 3 menunjukkan potongan kode fungsi `getAllBookings()` yang digunakan untuk mengambil seluruh data pesanan dari basis data dengan dukungan fitur pencarian dan *pagination*. Setelah data diperoleh, fungsi menghitung total data, jumlah halaman, dan menentukan apakah terdapat halaman selanjutnya atau sebelumnya. Hasil akhir berupa daftar pesanan dan *metadata* dikembalikan untuk ditampilkan pada antarmuka.

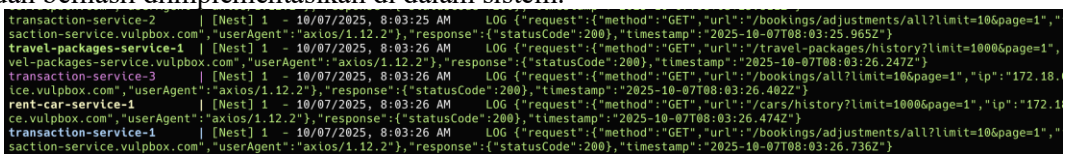
Kode 4. Potongan Kode Data Semua Pesanan

```
public async getAllBookings(
    paginationDto: PaginationDto
  ): Promise<BookingResDto> {
    try {
      const { page = 1, limit = 10, search = '' } = paginationDto;

      const queryBuilder = this.bookingRepository
        .createQueryBuilder('bookings')
        .leftJoinAndSelect('bookings.payments', 'payments')
        .orderBy('bookings.created_at', 'DESC')
        .addOrderBy('payments.created_at', 'DESC');
      const conditions = [];
      const parameters: Record<string, any> = {};
      if (search) {
        conditions.push('CAST(bookings.status AS TEXT) ILIKE :search');
        parameters['search'] = `%${search}%`;
      }

      if (conditions.length) {
        queryBuilder.where(conditions.join(' OR '), parameters);
      }
      const [result, total] = await queryBuilder
        .skip((page - 1) * limit)
        .take(limit)
        .getManyAndCount();

      const totalPages = Math.ceil(total / limit);

      const hasNextPage = page < totalPages;
      return {
        data: result,
        meta: {totalItems: total,currentPage: page,totalPages,limit,
          hasNextPage,hasPrevPage: page > 1};
    } catch (error) {throw new HttpException(error.message, error.status);}
  }
```

4.5 Pengujian Sistem

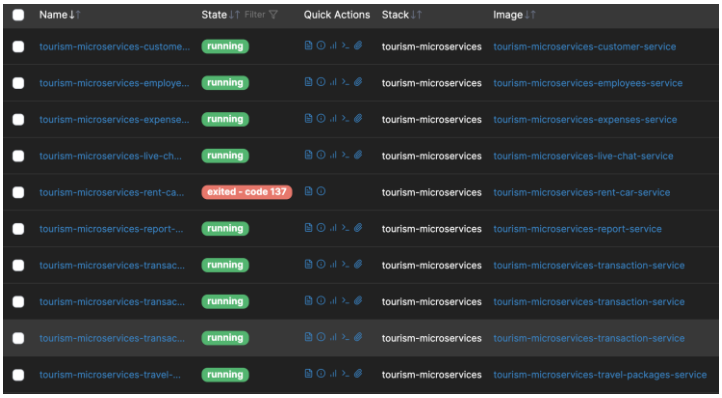
Pengujian sistem dibagi menjadi pengujian arsitektur, pengujian fungsionalitas menggunakan metode *Black Box testing* dan pengujian kuesioner terhadap pengguna. Pengujian arsitektur dilakukan untuk memastikan arsitektur dapat mengimplementasikan *scalability* dan juga *fault tolerance*. Pengujian fungsionalitas untuk memastikan fungsi-fungsi di dalam sistem berjalan lancar. Sedangkan pengujian menggunakan kuesioner untuk menilai sejauh mana sistem berjalan sesuai spesifikasi, serta mampu memenuhi kebutuhan dan harapan pengguna.

Pada Gambar 7 menampilkan bahwa *log request* ke layanan *transaction service* dibagi ke tiga *instance* yang berbeda. Pada *log* tersebut, *request* dengan *path* url `/bookings/adjustments/all?limit=10&page=1` diarahkan ke *instance transaction-service-2*, *request* selanjutnya diarahkan ke *transaction-service-3*, dan terakhir masuk ke *transaction-service-1*. Hal ini mengindikasikan bahwa *scalability* sistem dan *load balancing* sudah berhasil diimplementasikan di dalam sistem.

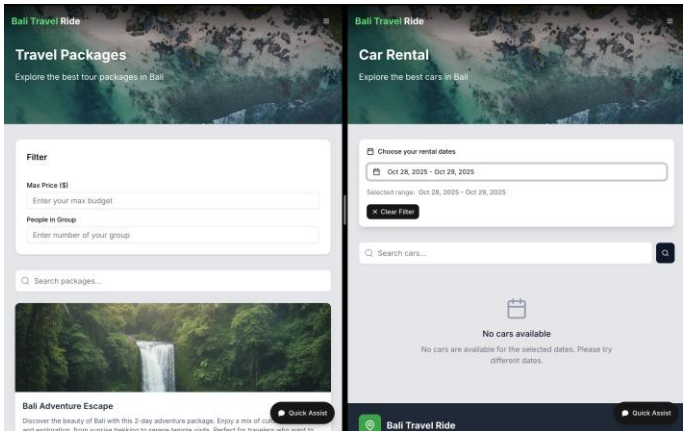


Gambar 7. *Log Request* Masuk Tiap Layanan *Microservices*

Pada Gambar 8 dan Gambar 9 menampilkan simulasi isolasi kesalahan pada sistem. Pada Gambar 8 menampilkan bahwa layanan *rent car service* dinonaktifkan, tetapi layanan lain tetap berjalan. Kemudian, pada Gambar 9 menampilkan halaman *travel packages* tetap menampilkan data *travel*, tetapi pada halaman *car rental* tidak menampilkan data sama sekali. Hal ini mengindikasikan bahwa meskipun ada layanan yang mengalami masalah, tidak akan mempengaruhi layanan lainnya. Hal ini membuktikan bahwa *fault tolerance* berhasil diimplementasikan di dalam sistem.



Gambar 8. Layanan Rent Car Service Dinonaktifkan



Gambar 9. Hasil Isolasi Kesalahan Layanan Pada Rent Car Service

Pengujian selanjutnya adalah melakukan pengujian pada fungsionalitas sistem untuk memastikan fungsi-fungsi di dalam sistem berjalan sesuai dengan yang diharapkan. Pengujian ini menggunakan metode Black Box *testing,* yaitu metode pengujian yang dilakukan pada sistem tanpa melihat kode program dari sistem. Dokumen pengujian dapat dilihat pada Tabel 1.

Tabel 1. Tabel Pengujian Black Box

Identifikasi	Kebenaran yang Diharapkan	Kesimpulan
Pemesanan Paket *Travel* dan pembayaran	Sistem akan mengarahkan ke halaman pembayaran dari *payment gateway*. Setelah pembayaran selesai, pengguna diarahkan ke halaman pembayaran berhasil.	Sesuai
Pemesanan Sewa Mobil dan pembayaran	Sistem akan mengarahkan ke halaman pembayaran dari *payment gateway*. Setelah pembayaran selesai, pengguna diarahkan ke halaman pembayaran berhasil.	Sesuai
Melihat histori pemesanan	Sistem menampilkan list pesanan yang pernah dipesan oleh pengguna	Sesuai
Mengajukan *Reschedule*	Sistem menampilkan pesan bahwa pengajuan *reschedule* berhasil dilakukan.	Sesuai
Mengajukan Pembatalan	Sistem menampilkan pesan bahwa pengajuan pembatalan berhasil dilakukan.	Sesuai
Mengkonfirmasi Pesanan baru	Sistem menampilkan pesan bahwa pesanan berhasil dikonfirmasi dan diarahkan ke halaman Data Pesanan	Sesuai
Menerima Pengajuan *Reschedule*	Sistem menampilkan pesan bahwa pengajuan *reschedule* berhasil diterima	Sesuai
Melihat Laporan	Pengguna diarahkanke halaman *submenu report* dan terampil data laporan pendapatan bulanan dan tahunan, laporan pengeluaran bulanan dan tahunan, serta laporan laba rugi.	Sesuai

Pengujian terakhir adalah pengujian kuesioner terhadap pengguna. Pengujian ini dilakukan untuk mengukur kemudahan penggunaan sistem, kelancaran pemesanan, kejelasan informasi, dan sejauh mana sistem mampu memenuhi kebutuhan dan harapan pengguna. Pengujian ini memiliki 32 responden per-pertanyaan dan hasil pengujian kuesioner dapat dilihat pada Tabel 2.

Tabel 2. Tabel Pengujian Terhadap Pelanggan

No	Pertanyaan	Jawaban				
		SS	S	C	TS	STS
1	Saya merasa mudah dalam mengakses serta berpindah halaman di *website* ini.	18	14	0	0	0
2	Saya merasa *website* ini memiliki tampilan yang menarik.	21	10	1	0	0
3	Saya merasa mudah menemukan dan memilih paket *travel* atau mobil yang saya inginkan.	16	16	0	0	0
4	Saya merasa informasi mengenai paket *travel* dan sewa mobil pada *website* ini ditampilkan dengan jelas.	17	13	2	0	0
5	Ulasan dari pelanggan lain membantu saya merasa lebih aman untuk melakukan pemesanan.	16	13	3	0	0
6	*Rating* memudahkan saya untuk membandingkan beberapa layanan secara cepat.	17	14	1	0	0
7	Saya merasa deskripsi ulasan dari pelanggan lain memberikan informasi tambahan yang penting sebelum saya memesan.	21	10	1	0	0
8	Saya merasa proses pemesanan paket *travel* dan sewa mobil terasa jelas dan lancar.	19	13	0	0	0
9	Metode pembayaran yang tersedia memberikan opsi yang saya butuhkan.	22	10	0	0	0
10	Proses pembayaran melalui aplikasi berjalan dengan lancar dan tanpa kendala.	21	10	0	1	0

No	Pertanyaan	Jawaban				
		SS	S	C	TS	STS
11	Halaman *'History Order'* pada *website* ini mudah diakses dan membantu saya dalam melihat pesanan yang telah saya buat.	23	9	0	0	0
12	Fitur '*Reschedule'* pada pesanan memberikan saya fleksibilitas untuk melakukan perubahan jadwal.	17	15	0	0	0
13	Fitur '*Cancellation'* & '*Refund'* pada pesanan memberikan saya rasa aman dan perlindungan dari risiko kerugian.	25	7	0	0	0
14	Saya merasa fitur '*Quick Assist'* memudahkan saya dalam mendapatkan bantuan secara cepat.	20	10	2	0	0
15	Saya merasa fitur '*Quick Assist'* membantu saya lebih memahami informasi di dalam *website* ini.	21	10	1	0	0

5. Kesimpulan dan Saran

Pembangunan *Website Paket Travel dan Sewa Mobil Berbasis Arsitektur Microservices* untuk Bali Travel Ride berhasil mengatasi kendala operasional yang sebelumnya dilakukan secara manual. Berdasarkan hasil Black Box *testing*, 80% responden yang terdiri dari pelanggan, admin, dan pemilik menyatakan bahwa sistem ini telah membantu operasional bisnis. Selain itu, penerapan arsitektur *microservices* terbukti efektif dalam mengatasi kompleksitas sistem karena setiap layanan dapat dikembangkan, dikelola, dan *deployed* secara independen tanpa mengganggu layanan lain namun tetap terintegrasi dengan baik.

Berdasarkan hasil penelitian dan pembangunan *website* paket *travel* dan sewa mobil berbasis *microservices* ini dan pertimbangan potensi pengembangan di masa mendatang, terdapat beberapa saran yang dapat diberikan untuk menyempurnakan fungsionalitas *website* ini, yaitu pengembangan fitur GPS pada mobil yang disewa dan terintegrasi dengan *website* berupa menampilkan peta letak tiap mobil dan pengembangan fitur *Chat Bot* pada *Live Chat*.

6. Ucapan Terima Kasih

Terima kasih saya sampaikan kepada Tuhan Yang Maha Esa atas rahmat dan karunia-Nya, kepada kedua orang tua tercinta atas doa dan dukungannya, pihak Bali Travel Ride atas kesempatan dan kerja sama yang diberikan dan telah membantu penelitian ini.

Referensi

[1] Badan Pusat Statistik Provinsi Bali, "Perkembangan Pariwisata Provinsi Bali April 2024," https://bali.bps.go.id/id/pressrelease/2024/06/03/717894/perkembangan-pariwisata-provinsi-bali-april-2024.html.

[2] Badan Pusat Statistik Provinsi Bali, "Perkembangan Pariwisata Provinsi Bali Mei 2024," https://bali.bps.go.id/id/pressrelease/2024/07/01/717895/perkembangan-pariwisata-provinsi-bali-mei-2024.html.

[3] M. G. Cristian, "An Overview On Tourism's Contribution To GDP," *Journal of economic-financial theory and practice*, vol. 72, no. 2, pp. 19–26, Jul. 2020, Accessed: Oct. 22, 2025. [*Online*]. Available: http://oldeconomice.ulbsibiu.ro/revista.economica/articol.php?id=1158

[4] J. Lei, L. Indiran, and U. H. A. Kohar, "Barriers to Digital Transformation among MSME in Tourism Industry: Cases Studies from Bali," *International Journal of Academic Research in Business and Social Sciences*, vol. 13, no. 3, pp. 802–816, Mar. 2023, doi: 10.6007/ijarbss/v13-i3/16575.

[5] R. Elsen and J. Algoritma, "Perancangan Arsitektur Microservice untuk Portal Berita Daring," *Jurnal Algoritma*, vol. 18, no. 2, pp. 352–357, 2021, doi: 10.33364/algoritma/v.18-2.875.

[6] C. Seviro, B. Sakti, and I. Hermawan, "Implementasi Arsitektur Microservice Pada Back End Sistem Informasi Antlantas Berbasis Web," *Jurnal Teknologi Terpadu*, vol. 6, no. 2, pp. 96–104, Dec. 2020, doi: 10.54914/jtt.v6i2.281.

[7] U. Reginal and S. D. Riskiono, "Sistem Informasi Pelanan Jasa Tour Dam Travel Berbasis Web (Studi Kasus Smart Tour)," *Jurnal Informasi Dan Komputer*, vol. 6, no. 2, pp. 51–62, Oct. 2018, doi: 10.35959/jik.v6i2.112.

[8] A. Anshori, A. H. Brata, and L. Fanani, "Pengembangan Sistem Reservasi Rental Kendaraan dan Trip Wisata berbasis Web (Studi Kasus: G19 Tour & Travel)," *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, vol. 7, no. 6, pp. 2729–2735, Jun. 2023, Accessed: Oct. 22, 2025. [*Online*]. Available: https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/12874

[9] M. Khathab and M. Rasyid Ridha, "Sistem Informasi Pemesanan Tiket Pada Indah Travel Berbasis Web," *Jurnal Perangkat Lunak*, vol. 2, no. 2, pp. 63–71, Jun. 2020, doi: 10.32520/jupel.v2i2.1100.

[10] C. A. Dinova and I. C. Utomo, "Pengembangan Arsitektur Microservice pada Learning Management System E-learning Menggunakan Metode Web Service," *Jurnal Ilmu Komputer dan Informatika*, vol. 3, no. 2, pp. 125–141, Apr. 2024, doi: 10.54082/jiki.102.

[11] A. Maspupah, "Literature Review: Advantages and Disadvantages Of *Black Box* and White Box Testing Method," *Jurnal Techno Nusa Mandiri*, vol. 21, no. 2, pp. 151–162, Sep. 2024, doi: 10.33480/techno.v21i2.5776.