Frontiers in Emerging Computer Science and Information Technology (Open Access)

Volume 02, Issue 01, January 2025,
Publish Date: 01-01-2025
PageNo.01-07

A Microservices-Driven Framework for Integrated Travel Data and Service Delivery

Prof. George Papadopoulos
Department of Computer Science, University of Cyprus, Nicosia, Cyprus

Prof. Andreas Antoniou
Department of Computer Science, University of Cyprus, Nicosia, Cyprus

ABSTRACT

The contemporary travel industry operates on a vast, intricate network of data sources, ranging from flight availability and hotel bookings to car rentals and personalized itinerary preferences. Traditional monolithic architectural paradigms often struggle to cope with the dynamic nature, immense scale, and diverse integration requirements of this sector, leading to scalability bottlenecks, slow innovation cycles, and fragmented user experiences. This article proposes a novel microservices-driven framework for seamless travel data integration and enhanced service delivery. By decomposing the monolithic travel platform into a collection of independently deployable, loosely coupled, and specialized microservices, the architecture aims to overcome the limitations of conventional systems. Each microservice is designed to manage specific business capabilities (e.g., flight search, hotel booking, payment processing, user profiles), fostering agility, scalability, and resilience. The framework emphasizes distributed data management, efficient inter-service communication, and cloud-native deployment strategies to ensure high availability and responsiveness. We detail the architectural components, data flow, and operational advantages, demonstrating how this approach can facilitate real-time data aggregation, enable personalized travel experiences, and accelerate the development and deployment of new features, thereby revolutionizing the landscape of online travel service provision.

KEYWORDS: Microservices architecture, integrated travel data, service delivery, travel platforms, API integration, distributed systems, cloud computing, travel technology, scalable frameworks, real-time data processing.

INTRODUCTION

The global travel industry is a multifaceted and rapidly evolving sector, characterized by its dynamic pricing, vast inventory, complex reservation systems, and ever-increasing demand for personalized services [13]. From flight booking and hotel reservations to car rentals, tour packages, and ancillary services, the digital travel ecosystem relies heavily on robust and efficient information technology infrastructure. Historically, many online travel agencies (OTAs) and direct service providers have relied on monolithic architectural designs, where all functionalities are tightly coupled within a single, large application [3], [39]. While simple to develop initially, these monolithic systems quickly become unwieldy as they scale, leading to significant challenges in terms of scalability, maintainability, agility, and resilience [6], [11], [14], [27].

The limitations of monolithic architectures in the travel domain are manifold. Scaling a monolithic application often means scaling the entire system, even if only a small component is experiencing high load, leading to inefficient resource utilization. Deploying new features or bug fixes requires redeploying the entire application, resulting in prolonged downtime and slow development cycles. Furthermore, the tightly coupled nature of components makes the system fragile; a failure in one part can cascade and bring down the entire application, jeopardizing service availability and customer satisfaction [5]. Integrating new third-party APIs (e.g., for new airlines, hotels, or payment gateways) or adopting new technologies becomes a complex and time-consuming endeavor [27].

In response to these challenges, the microservices architecture has emerged as a powerful paradigm for building modern, scalable, and resilient software systems [1], [3], [8], [11], [12], [14], [27]. Microservices promote the decomposition of a large application into a collection of small, independent services, each responsible for a specific business capability, running in its own process, and communicating via lightweight mechanisms, typically APIs [1], [3]. This architectural style offers significant advantages, including enhanced scalability, improved fault isolation, accelerated development cycles, and increased technological flexibility [1], [3], [5], [12], [27]. Its principles are particularly well-suited to complex, data-intensive, and highly dynamic domains like the travel industry [9], [17], [26], [28], [38].

The fragmented nature of travel data presents another significant hurdle. Information about flights, accommodations, pricing, and availability resides across numerous disparate systems and providers. Effectively integrating and aggregating this distributed travel data in real-time, while simultaneously ensuring data consistency and providing seamless service delivery, is crucial for offering competitive and comprehensive travel solutions. Traditional integration patterns in monolithic systems often lead to complex, brittle dependencies and data silos [9].

This article proposes a novel architectural framework that leverages the microservices paradigm to facilitate robust, distributed travel data integration and advanced service provision. Our primary objective is to detail an architectural design that overcomes the limitations of monolithic systems by enabling scalable data aggregation, enhancing service responsiveness, and fostering agility in feature development for the online travel domain. We hypothesize that a microservices-driven approach will significantly improve system performance, resilience, and adaptability, paving the way for next-generation online travel platforms capable of delivering highly personalized and real-time user experiences.

The remainder of this paper is structured as follows: Section 2 outlines the methodology, detailing the proposed microservices architecture, its key components, data integration strategies, and the underlying technological considerations. Section 3 discusses the expected results and advantages of implementing such a framework. Section 4 provides a comprehensive discussion of the implications, challenges, and future directions. Finally, Section 5 concludes the article.

METHODS

Architectural Overview: The Microservices Paradigm for Travel

The proposed framework embraces the microservices architectural style to address the complexities of travel data integration and service provision. Instead of a single, monolithic application, the online travel platform is decomposed into a set of small, autonomous services, each owning its specific business domain and data. This decomposition follows the principle of "bounded contexts," ensuring that each service has a clear responsibility and minimal dependencies on others [3]. The overall system architecture is depicted conceptually in Figure 1.

(This would typically be an actual figure in an academic paper, illustrating the flow: User Interface -> API Gateway -> various independent Microservices (e.g., Flight Service, Hotel Service, Payment Service, User Profile Service, Recommendation Service), each with its own database, possibly interacting via Message Queues. Cloud infrastructure, containers, and orchestration would be implied.)

Core Principles and Service Decomposition

The design adheres to fundamental microservices principles [1]:

- Loose Coupling: Services are designed to be independent, minimizing direct dependencies. Changes to one service do not necessitate changes or redeployments of others.
- High Cohesion: Each service encapsulates a single, well-defined business capability.
- Independent Deployability: Services can be developed, tested, and deployed independently, accelerating release cycles.
- Data Ownership: Each service manages its own data store, promoting autonomy and avoiding shared database anti-patterns [9].
- Decentralized Governance: Different teams can choose the best technology stack for their specific service, fostering innovation.

Based on typical travel platform functionalities, the system is decomposed into a set of core microservices, including but not limited to:

- User Management Service: Handles user authentication, authorization, and profile management.
- Flight Service: Manages flight search, booking, real-time availability, and airline integrations [17], [22], [26].
- Hotel Service: Manages hotel search, reservations, room availability, and hotel chain integrations.
- Car Rental Service: Facilitates car search and booking across various providers.
- Package/Tour Service: Aggregates combinations of flights, hotels, and activities.
- Payment Service: Securely processes transactions, integrates with various payment gateways.
- Booking Management Service: Manages post-booking operations, cancellations, modifications.
- Notification Service: Handles communication with users (email, SMS notifications).
- Recommendation Service: Provides personalized travel recommendations based on user behavior and preferences, potentially leveraging AI/ML [7], [26], [33], [34].
- Review/Rating Service: Manages user-generated content for destinations, hotels, flights.

Data Integration Strategy

The key challenge in distributed travel systems is integrating data from numerous heterogeneous sources while maintaining consistency. The framework adopts a decentralized data management approach where each microservice owns its domain data [9].

- Polyglot Persistence: Services can use the database technology best suited for their specific data needs. For instance, the User Management service might use a relational database like PostgreSQL for structured user data [20], [21], [25], while the Recommendation service might use a NoSQL database optimized for large-scale key-value or graph data.
- API-driven Data Access: Services expose their data through well-defined APIs. Other services requiring this data must call the respective API, ensuring encapsulation and controlled access [10].
- Event-Driven Architecture: For cross-service data synchronization and complex workflows, an event-driven architecture with message brokers (e.g., Kafka, RabbitMQ) is employed [2]. When a significant event occurs within one service (e.g., "flight booked," "payment successful"), it publishes an event to a message queue. Other interested services can subscribe to these events and react accordingly, ensuring eventual consistency across the system without tight coupling. For example, a successful payment event from the Payment Service could trigger the Booking Management Service to finalize the booking.
- Data Aggregation at the Edge: For user-facing views requiring data from multiple services (e.g., a "My Trips" dashboard), an API Gateway acts as a single entry point for all client requests [2]. It routes requests to the appropriate microservices and can aggregate responses from multiple services into a single response for the client, masking the underlying complexity of the microservices architecture [2].

Technology Stack and Deployment

The framework relies on cloud-native technologies for efficient deployment and operation:

- Containerization (Docker): Each microservice is packaged into a lightweight, portable Docker container, ensuring consistent environments across development, testing, and production [4].
- Container Orchestration (Kubernetes): Kubernetes is utilized for automated deployment, scaling, and management of the containerized microservices across a cluster of servers [4], [31]. This provides self-healing capabilities, load balancing, and efficient resource utilization [5], [17], [31].
- Cloud Platform: Deployment on a robust cloud platform (e.g., Google Cloud Platform) provides the necessary infrastructure, elasticity, and global reach required by online travel systems [4], [17], [31], [38]. Serverless computing options (e.g., AWS Lambda, Google Cloud Functions) could also be integrated for specific functions, further enhancing scalability and cost-efficiency [2].
- Service Mesh: A service mesh (e.g., Istio, Linkerd) could be integrated to handle inter-service communication complexities such as traffic management, security, and observability in a distributed environment.
- Monitoring and Logging: Centralized logging and monitoring tools (e.g., Prometheus, Grafana, ELK stack) are essential for tracking the health, performance, and behavior of individual services in a distributed system.

Service Communication and API Management

- RESTful APIs: Services primarily communicate via synchronous RESTful APIs for request-response patterns. Each service exposes a well-documented API [10].
- Asynchronous Messaging: For event-driven flows and long-running processes, asynchronous messaging queues are used [2].
- API Versioning: A robust API versioning strategy is crucial to allow services to evolve independently without breaking clients or other services [10].

Scalability and Resilience Mechanisms

The microservices approach inherently builds in scalability and resilience:

- Horizontal Scaling: Individual services can be scaled independently based on demand, allowing efficient allocation of resources where needed [5].
- Fault Isolation: A failure in one microservice is contained and does not propagate to other services, ensuring the overall system's stability [5]. Load balancing and circuit breakers prevent overloaded services from causing cascading failures.
- Service Discovery: Services dynamically find and communicate with each other through a service registry, eliminating hardcoded network locations [5].

Results

Implementing a microservices-driven framework for travel data integration and service delivery is expected to yield several significant advantages over traditional monolithic architectures. These benefits directly address the challenges currently faced by the rapidly evolving online travel industry.

Enhanced Scalability

The fine-grained decomposition of the system into independent microservices enables unparalleled scalability

[5]. Each service can be scaled horizontally based on its specific workload demands, independent of other services. For instance, during peak travel seasons, the "Flight Search Service" can be scaled up to handle millions of queries per second without requiring the scaling of less-utilized services like "User Profile Management." This efficient resource utilization leads to better performance under high load and reduced operational costs compared to scaling an entire monolith [17]. Cloud-native deployment with Kubernetes further automates this process, ensuring elastic scaling and optimal resource allocation [4], [31].

Increased Resilience and Fault Isolation

One of the most critical advantages is the improved resilience of the system [5]. In a monolithic architecture, a single point of failure can lead to a complete system outage. With microservices, if one service experiences an issue or crashes, it is isolated, and other services can continue to operate normally. For example, if the "Car Rental Service" experiences a temporary outage, users can still book flights and hotels. This fault isolation significantly enhances the overall system's uptime and reliability, crucial for a 24/7 global industry like travel. Mechanisms like circuit breakers and bulkheads further prevent cascading failures.

Accelerated Development and Deployment Cycles

Microservices facilitate parallel development by independent teams, each focusing on a specific service [3]. This modularity significantly accelerates the development lifecycle, allowing teams to iterate, test, and deploy new features or bug fixes more frequently and with greater confidence [12]. The ability to deploy individual services without affecting the entire system minimizes downtime and reduces the risk associated with large-scale releases [3]. This agility allows travel platforms to respond rapidly to market changes, competitive pressures, and evolving customer demands.

Improved Data Consistency and Management

By promoting data ownership at the service level, the architecture inherently improves data consistency within each bounded context and simplifies data management [9]. While eventual consistency across services is often a design trade-off, this model avoids the complexities and contention issues of a large, shared monolithic database. Each service can choose the most appropriate database technology (polyglot persistence) for its specific data needs, leading to optimized performance and simplified schema evolution for individual data stores [9], [20], [21], [25].

Flexibility and Technological Agility

The decentralized nature of microservices fosters technological flexibility [27]. Teams can select the best programming languages, frameworks, and tools for each individual service, rather than being locked into a single technology stack [1], [3]. This allows the platform to adopt new innovations rapidly (e.g., integrate new AI/ML models [7], [22], [26], [28], [33], [34], [35], [36], [37] or blockchain for trust [32]) and integrate with third-party APIs more easily, providing a competitive edge.

Enhanced Personalization and Real-time Capabilities

The microservices architecture, particularly with an event-driven backbone, is exceptionally well-suited for enabling advanced personalization and real-time data processing. Dedicated microservices for "Recommendation" or "Dynamic Pricing" can leverage real-time user behavior data and external market feeds to offer tailored travel options and pricing, significantly enhancing the customer experience and optimizing revenue [7], [28], [34], [37]. The ability to process data at the edge can further enhance responsiveness for certain use cases [36].

DISCUSSION

The implementation of a microservices-driven framework for travel data integration and service delivery represents a paradigm shift from traditional monolithic systems, offering substantial advantages in an increasingly complex and competitive industry. The presented architecture inherently addresses the critical needs for scalability, resilience, and agility that are often lacking in legacy travel platforms [6], [27], [39].

The modularity of microservices facilitates the continuous innovation demanded by the travel sector. For instance, the ability to independently develop and deploy a new "AI-driven Itinerary Optimization Service" [33] or a "Blockchain-based Loyalty Program Service" [32] without disrupting the core booking functionalities is a testament to this architectural flexibility. This allows travel companies to quickly prototype, test, and roll out new features, maintaining a competitive edge and responding to evolving customer preferences for personalized and seamless travel experiences [26], [34].

Despite these compelling advantages, the adoption of a microservices architecture is not without its challenges. The primary hurdle lies in the increased operational complexity [6], [27]. Managing a distributed system with numerous independently deployed services requires robust infrastructure, sophisticated monitoring, centralized logging, and automated deployment pipelines [4], [31]. Debugging issues across multiple services can be significantly more challenging than in a monolithic application. Strategies for effective service discovery, load balancing, and fault tolerance become critical [5].

Data consistency across services, while managed by individual service ownership, transitions from immediate consistency (typical in monoliths with shared databases) to eventual consistency, which requires careful design to avoid user experience issues [9]. Developers must adopt new paradigms like sagas or distributed transactions where strong consistency is required, adding complexity. Furthermore, API management and versioning are crucial to ensure backward compatibility and prevent breaking changes as services evolve [10]. Without a disciplined approach, the benefits of independent deployability can be negated by inter-service dependency hell.

Security in a microservices environment also presents unique considerations, as security measures need to be applied at the service level, and communication channels between services must be secured [27]. Finally, the cultural shift within development teams is significant; moving from a single large team working on a monolith to multiple small, autonomous teams managing distinct services requires new organizational structures and communication strategies.

Future Work

The proposed microservices framework opens several exciting avenues for future research and development within the travel domain:

- Deeper AI/ML Integration: Further research into integrating advanced AI and machine learning models directly into microservices for enhanced personalization [7], dynamic pricing [34], real-time recommendations [26], [33], and predictive analytics for operational efficiency [28], [37]. This includes exploring AI-driven resource allocation for microservices within hybrid cloud environments to optimize performance and cost [37].
- Blockchain for Trust and Transparency: Investigate the practical implementation of blockchain technology within specific microservices (e.g., for secure payment processing, verifiable loyalty programs, or immutable booking records) to enhance trust and transparency across the travel ecosystem [32], [35].
- Edge Computing Integration: Explore the deployment of lightweight microservices or edge-enabled components closer to the data sources or end-users to enable real-time processing and ultra-low latency for critical operations, such as immediate availability checks or personalized alerts [24], [36].
- Serverless Microservices: Further explore the benefits and challenges of deploying individual microservices as serverless functions (Function-as-a-Service, FaaS) to achieve extreme scalability and reduce operational overhead, particularly for event-driven workflows [2].
- Performance Optimization: Conduct empirical studies on the real-time performance optimization of such systems, including strategies for caching, data replication, and distributed tracing to identify and resolve bottlenecks in complex microservice interactions [37].
- Automated Governance and Observability: Develop more sophisticated tools and methodologies for automated governance, monitoring, and self-healing capabilities within large-scale microservices deployments to manage their inherent complexity.
- Security Frameworks: Design and evaluate comprehensive security frameworks specifically tailored for distributed microservices in the travel industry, considering aspects like API security, data encryption, and access control across services.

In conclusion, the microservices-driven framework for integrated travel data and service delivery offers a robust, scalable, and agile solution for the modern travel industry. By embracing this architectural paradigm, travel platforms can overcome the limitations of legacy systems, foster rapid innovation, and deliver superior, highly personalized experiences to travelers worldwide. While challenges associated with distributed systems remain, continuous advancements in cloud computing, container orchestration, and AI/ML will further solidify the microservices approach as the de facto standard for future-proof travel technology solutions.

REFERENCES

[1] Lewis J, Fowler M: Microservices: A definition of this new architectural term. (2014). https://martinfowler.com/articles/microservices.html.

[2] Barua B, Kaiser MS: A methodical framework for integrating serverless cloud computing into microservice architectures. Preprints. 10.20944/preprints202410.0494.v1.

[3] Newman S: Building microservices: Designing fine-grained systems. (2021). https://book.northwind.ir/bookfiles/building-microservices/Building.Microservices.pdf.

[4] Shah J, Dubaria D: Building modern clouds: Using Docker, Kubernetes & Google Cloud Platform. 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC). 2019, 0184-0189. 10.1109/CCWC.2019.8666479.

[5] Barua B, Kaiser MS: Enhancing resilience and scalability in travel booking systems: A microservices approach to fault tolerance, load balancing, and service discovery. 10.48550/arXiv.2410.19701.

[6] Jamshidi P, Pahl C, Mendonça NC, Lewis J, Tilkov S: Microservices: The journey so far and challenges ahead. IEEE Software. 2018, 35:24-35. 10.1109/ms.2018.2141039.

[7] Barua B, Kaiser MS: Leveraging machine learning for real-time personalization and recommendation in airline industry [PREPRINT]. 10.20944/preprints202410.2436.v1.

[8] Thönes J: Microservices. IEEE Software. 2015, 32:116-116. 10.1109/ms.2015.11.
[9] Barua B, Whaiduzzaman M, Sarker M, Kaiser M, Barros A: Designing and implementing a distributed database for microservices cloud-based online travel portal. Sentiment Analysis and Deep Learning. Shakya S, Du KL, Ntalianis K (ed): Springer, Singapore; 2023. 1432:295-314. 10.1007/978-981-19-5443-6_22.
[10] Serbout S, Pautasso C: How are web APIs versioned in practice? A large-scale empirical study. Journal of Web Engineering. 2024, 23:465-506. 10.13052/jwe1540-9589.2341.
[11] Dragoni N, Giallorenzo S, Lluch Lafuente A, Mazzara M, Montesi F, Mustafin R, Safina L: Microservices: Yesterday, today, and tomorrow. Present and Ulterior Software Engineering. Mazzara M, Meyer B (ed): Springer, Cham; 2017. 195-216. 10.1007/978-3-319-67425-4_12.
[12] Bakshi K: Microservices-based software architecture and approaches. 2017 IEEE Aerospace Conference, Big Sky, MT, USA. 2017, 1-8. 10.1109/AERO.2017.7943959.
[13] Barua B: M-commerce in Bangladesh -status, potential and constraints. International Journal of Information Engineering and Electronic Business. 2016, 8:22-27. 10.5815/ijieeb.2016.06.03.
[14] Larrucea X, Santamaria I, Colomo-Palacios R, Ebert C: Microservices. IEEE Software. 2018, 35:96-100. 10.1109/ms.2018.2141030.
[15] Barua B, Whaiduzzaman M: A methodological framework on development the garment payroll system (GPS) as SaaS. 2019 1st International Conference on Advances in Information Technology (ICAIT). 2019, 431-435. 10.1109/ICAIT47043.2019.8987325.
[16] Howlader SMN, Barua B, Sarker MMS, Kaiser MS, Whaiduzzaman M: Automatic yard monitoring and humidity controlling system based on IoT. 2023 International Conference on Advanced Computing & Communication Technologies (ICACCTech). 2023, 397-403. 10.1109/ICACCTech61146.2023.00072.
[17] Barua B, Kaiser MS: Cloud-enabled microservices architecture for next-generation online airlines reservation systems [PREPRINT]. 10.21203/rs.3.rs-5182678/v1.
[18] Barua B, Obaidullah MD: Development of the student management system (SMS) for universities in Bangladesh. BUFT Journal. 2014, 2:57-66.
[19] Chaki PK, Sazal MMH, Barua B, Hossain MS, Mohammad KS: An approach of teachers' quality improvement by analyzing teaching evaluations data. 2019 Second International Conference on Advanced Computational and Communication Paradigms (ICACCP). 2019, 1-5. 10.1109/ICACCP.2019.8882915.
[20] Juba S, Volkov A: Learning PostgreSQL 11: A Beginner's Guide to Building High-Performance PostgreSQL Database Solutions. Packt, Birmingham, UK; 2019.
[21] Momjian B: PostgreSQL: Up and Running (4th ed). O'Reilly Media, Sebastopol, CA; 2021.
[22] Barua B, Mozumder MJU, Kaiser MS, Barua I: Trends and challenges in AI-driven microservices for cloud-based airline reservation systems: A review. 2025 3rd International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT). 2025, 1902-1911. 10.1109/IDCIOT64235.2025.10915076.
[23] Kubra KT, Barua B, Sarker MM, Kaiser MS: An IoT-based framework for mitigating car accidents and enhancing road safety by controlling vehicle speed. 2023 7th International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC). 2023, 46-52. 10.1109/I-SMAC58438.2023.10290219.
[24] Tian H, Xu X, Lin T, Cheng Y, Qian C, Ren L, Bilal M: DIMA: Distributed cooperative microservice caching for internet of things in edge computing by deep reinforcement learning. World Wide Web. 2022, 25:1769-1792. 10.1007/s11280-021-00939-7.
[25] Ünal HT, Mete S, Vurgun OU, Mendi AF, Özkan Ö, Nacar MA: Postgresql database management system: ODAK. 2023 Innovations in Intelligent Systems and Applications Conference (ASYU). 2023, 1-5. 10.1109/ASYU58738.2023.10296600.
[26] Barua B, Mozumder MJU, Kaiser MS, Barua I: Building scalable airlines reservation systems: A microservices approach using AI and deep learning for enhanced user experience. 2025 4th International Conference on Sentiment Analysis and Deep Learning (ICSADL). 2025, 941-950. 10.1109/ICSADL65848.2025.10933362.
[27] Di Francesco P, Lago P, Malavolta I: Architecting with microservices: A systematic mapping study. Journal of Systems and Software. 2019, 150:77-97. 10.1016/j.jss.2019.01.001.
[28] Barua B, Kaiser MS: Microservices-based framework for predictive analytics and real-time performance enhancement in travel reservation systems. 10.48550/arXiv.2412.15616.
[29] Barua B, Kaiser MS: Novel architecture for distributed travel data integration and service provision using microservices. 10.48550/arXiv.2410.24174.
[30] Chaki PK, Barua B, Sazal MMH, Anirban S: PMM: A model for Bangla parts-of-speech tagging using sentence map. Information, Communication and Computing Technology. Badica C, Liatsis P, Kharb L, Chahal D (ed): Springer, Singapore; 2020. 1170:181-194. 10.1007/978-981-15-9671-1_15.
[31] Barua B, Kaiser MS: A methodological framework of containerized microservices orchestration and provisioning in the cloud: A case study of online travel platforms. Challenges and Opportunities for Innovation in India. Mishra S, Singh AK, Prajapati P (ed): CRC Press, London; 2025. 183. 10.1201/9781003606260-33.

[32] Barua B, Kaiser MS: Blockchain-based trust and transparency in airline reservation systems using microservices architecture. 10.48550/arXiv.2410.14518.
[33] Barua B, Kaiser MS: Optimizing travel itineraries with AI algorithms in a microservices architecture: Balancing cost, time, preferences, and sustainability. 10.48550/arXiv.2410.17943.
[34] Barua B, Kaiser MS: Leveraging microservices architecture for dynamic pricing in the travel industry: Algorithms, scalability, and impact on revenue and customer satisfaction. 10.48550/arXiv.2411.01636.
[35] Barua B, Kaiser MS: A next-generation approach to airline reservations: Integrating cloud microservices with AI and blockchain for enhanced operational performance. 10.48550/arXiv.2411.06538.
[36] Barua B, Kaiser MS: Optimizing airline reservation systems with edge-enabled microservices: A framework for real-time data processing and enhanced user responsiveness. 10.48550/arXiv.2411.12650.
[37] Barua B, Kaiser MS: AI-driven resource allocation framework for microservices in hybrid cloud platforms. 10.48550/arXiv.2412.02610.
[38] Barua B, Kaiser MS: Real-time performance optimization of travel reservation systems using AI and microservices. 10.48550/arXiv.2412.06874.
[39] Newman S: Monolith to Microservices: Evolutionary Patterns to Transform Your Monolith. O'Reilly Media, Sebastopol, CA; 2021.
[40] Howlader SMN, Hossain MM, Khanom S, Sarker S, Barua B: An intelligent car locating system based on Arduino for a massive parking place. Multi-Strategy Learning Environment. Vimal V, Perikos I, Mukherjee A, Piuri V (ed): Springer, Singapore; 2024. 19-33. 10.1007/978-981-97-1488-9_2.
41] Barua B, Kaiser MS: Design and evaluation of a microservices cloud framework for online travel platforms. 10.48550/arXiv.2505.14508.