Intelligent Cloud-Native Car Rental Platforms: A Review of CRM-Centric Architectures, Conversational Booking, Digital KYC, and Concurrency-Safe Reservation Systems

Om Taskar[1], Sanket Timande[2], Khushi Jambhulkar[3], Diksha Kumbhare[4], Pooja Jaiswal[5]

[1,2,3]UG Student, Department of CSE (Data Science), Guru Nanak Institute of Technology, Nagpur
[4]Assistant Professor (Guide), Department of CSE (Data Science), Guru Nanak Institute of Technology, Nagpur
[5]Head of Department, Department of CSE (Data Science), Guru Nanak Institute of Technology, Nagpur

Email: [1]omtaskar7@gmail.com, [2]sankettimande7@gmail.com, [3]khushijambhulkar9@gmail.com

---

Abstract - Car rental platforms are evolving from counter-based operations and stand-alone booking websites into integrated, cloud-native systems that combine customer self-service with enterprise fleet governance. This review surveys the technologies and research that underpin such platforms. We organise the literature into eight themes: cloud and CRM platforms, conversational and voice interfaces, pricing transparency and revenue management, digital identity verification and privacy, concurrency control for reservations, emergency response and telematics, customer service recovery and sentiment, and governance, auditability, and business intelligence. For each theme we summarise the foundational work, the techniques used in practice, and the trade-offs that affect rental systems. We then compare architectural approaches and concurrency-control strategies, and use Homeride, a Salesforce-based car rental platform, as a reference case to show how the themes combine in a single implementation. The review identifies open challenges, including multilingual voice understanding for Indian users, legally compliant e-KYC after the 2018 Aadhaar judgment and the Digital Personal Data Protection Act, 2023, race-safe booking under API and bulk traffic, connectivity-independent emergency alerts, and the lack of empirical evaluations of integrated platforms. We conclude with research directions for building secure, transparent, and user-centred mobility platforms.

Key Words: Car rental, shared mobility, cloud computing, customer relationship management, conversational AI, voice user interface, e-KYC, data protection, concurrency control, telematics, survey.

1. INTRODUCTION

Shared mobility, including car rental, car-sharing, and on-demand services, has expanded rapidly, driven largely by online reservation and digital fleet management [1]. In India, demand for self-drive rentals has grown alongside smartphone adoption, yet many operators still verify documents at a counter, track fleets in spreadsheets, and run booking websites that are disconnected from accounting and customer service. These gaps cause identity fraud, abandoned bookings, double reservations, slow emergency response, and limited management visibility.

Modern platforms address these problems by combining several technologies: multi-tenant cloud platforms and CRM systems [2], [3], conversational interfaces [4], transparent pricing [5], digital identity verification [6], transactional concurrency control [7], and location-aware emergency services [8]. Research on each topic is extensive but scattered across databases, human–computer interaction, information systems, transportation, and law. There is little consolidated guidance on how these pieces fit together in a rental platform.

This paper reviews that body of work with three objectives: (i) to organise the relevant research into a taxonomy specific to intelligent car rental platforms; (ii) to compare architectural and technical approaches and their trade-offs; and (iii) to identify open problems, especially in the Indian regulatory context. We use Homeride, a car rental platform built on Salesforce Experience Cloud and the Lightning Platform, as a running reference case.

1.1 Review Methodology

The review follows the planning, conducting, and reporting stages recommended for structured literature reviews in software engineering [9]. Sources were drawn from peer-reviewed journals and conference proceedings, standards bodies (NIST, W3C, OWASP), official platform documentation, industry usability research, and primary Indian legal texts. Inclusion criteria were relevance to at least one platform capability in the taxonomy, availability of a verifiable published source, and, for technical documentation, currency with the platform version in use. Because several themes (for example, Apex locking semantics and Indian data protection law) are defined by documentation and statute rather than academic papers, these primary sources are included alongside scholarly work. The review is therefore integrative rather than a quantitative meta-analysis.

1.2 Contributions

1) An eight-theme taxonomy of the capabilities of intelligent cloud-native car rental platforms (Fig. 1).
2) A thematic synthesis of foundational and applied work for each theme, with the trade-offs relevant to rental operations.
3) Comparative analyses of reservation concurrency strategies (Table 1) and platform architectures (Table 2).
4) A mapping of the themes to a reference implementation (Table 3) and a set of open challenges and research directions.

2. BACKGROUND

2.1 Cloud Computing and Multi-Tenant PaaS

NIST defines cloud computing through on-demand self-service, broad network access, resource pooling, rapid elasticity, and measured service, delivered as IaaS, PaaS, or

SaaS [2]. Multi-tenant PaaS platforms such as Force.com store each tenant's schema as metadata over shared physical tables and enforce *governor limits* on queries, DML, and CPU time per transaction to protect shared resources [3], [10]. For application developers this removes infrastructure management but imposes a bulk-safe programming discipline.

2.2 Customer Relationship Management

CRM systems centralise customer identities, interactions, and service cases. When a rental platform is built natively on a CRM, bookings, payments, reviews, and incidents become related records of the customer, enabling unified service views, workflow automation, and reporting without integration middleware.

2.3 Two-Sided Platforms

Rochet and Tirole characterise platforms that serve two distinct groups whose participation is interdependent [11]. A rental platform is two-sided in an operational sense: customers and operators interact through the same records but require different interfaces, permissions, and workflows. Role-based access control (RBAC) provides the standard model for separating these concerns [12].

3. TAXONOMY OF PLATFORM CAPABILITIES

Fig. 1 presents the taxonomy used to structure this review. The eight themes cover the customer journey from discovery (conversational and voice interfaces, pricing transparency), through commitment (identity verification, reservation integrity), to operation (emergency response, service recovery) and management (governance and BI), all resting on the cloud and CRM platform layer.

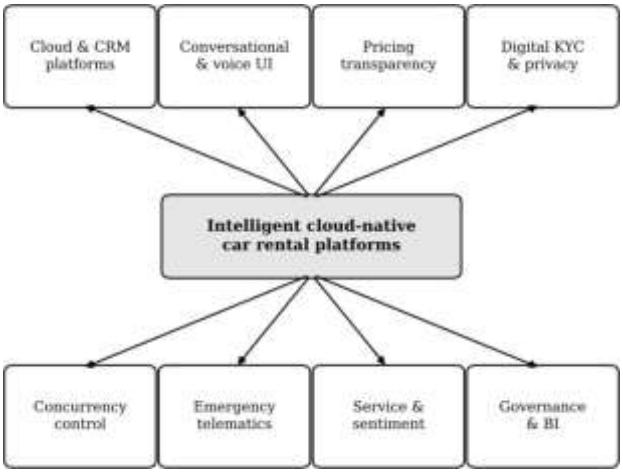


Fig -1: Taxonomy of capabilities in intelligent cloud-native car rental platforms.

4. THEMATIC REVIEW

4.1 Cloud and CRM Platform Architectures

Three architectural patterns appear in practice. *Counter-centric* operations rely on manual processes with minimal software. *Stand-alone web stacks* build a custom booking site on a general-purpose framework and database, then integrate payments, e-mail, and support tools separately. *Platform-native* systems build the storefront and back office on one PaaS with a shared data model. The metadata-driven design of Force.com illustrates the platform-native approach: tenants obtain a relational schema, transactional semantics, security model, and automation engine as services [3]. Layered architectures that separate presentation, business logic, automation, and persistence remain the dominant structuring principle, and object-oriented patterns such as the trigger-handler (a variant of the command and strategy patterns) help keep logic testable [13], [14]. The principal trade-offs are vendor lock-in and governor limits on the platform side versus integration and operational cost on the custom side.

4.2 Conversational and Voice Interfaces

Chatbots have progressed from pattern-matching systems such as ELIZA through rule-based and retrieval systems to neural sequence models and large language models [4]. For task-oriented booking, the critical steps are intent detection and slot filling, in which named entity recognition maps utterances to values such as locations, dates, and vehicle classes [15], [16]. Rule- and dictionary-based NER remains attractive for narrow domains because it is predictable and requires no training data, while statistical and neural approaches generalise better to varied phrasing [15]. On the web, the Web Speech API exposes browser speech recognition and synthesis to JavaScript [17], enabling voice input in a standard storefront without native apps. Usability research emphasises that such modalities should reduce effort and errors rather than add novelty [18]. Open issues for India include code-mixed Hindi–English utterances, regional accents, and the privacy implications of cloud-based speech processing.

4.3 Pricing Transparency and Revenue Management

Checkout usability research by the Baymard Institute places average documented online cart abandonment at roughly 70% and identifies unexpected extra costs as a leading avoidable cause [5]. For rentals, taxes, refundable deposits, and coupon rules are frequent sources of surprise. Showing a complete, itemised estimate before collecting personal details is therefore a practical design response. On the supply side, rental revenue management treats fleet capacity as a perishable resource; Geraghty and Johnson report significant gains from systematic pricing and fleet planning at National Car Rental [19]. Combining transparent estimates with dynamic pricing raises a design tension: prices must change with demand yet remain predictable and explainable to customers.

4.4 Digital Identity Verification and Privacy

Rental operators must confirm that drivers hold a valid licence under the Motor Vehicles Act [20] and often request additional identity proof. Digital KYC can catch format errors, expired documents, and duplicates at submission. Aadhaar numbers carry a Verhoeff check digit [21], enabling offline detection of most typing errors. However, the legal context restricts what private operators may do. The Aadhaar Act, 2016 governs use of the number [22], and in *Puttaswamy* (2018) the Supreme Court struck down the provision that had allowed private entities to require Aadhaar authentication [23]. Consent-based alternatives, such as UIDAI's paperless offline e-KYC [6], allow a resident to share signed, minimal demographic data. The Digital Personal Data Protection Act, 2023 adds obligations on consent, purpose limitation, and data minimisation [24]. Platform designs therefore increasingly store only masked identifiers (for example, the last four Aadhaar digits), restrict KYC fields to verification staff, and separate format validation from ownership verification. Zero-trust principles, which require continuous verification of every request regardless of network location, complement these controls [25].

4.5 Concurrency Control for Reservations

Double booking is an instance of the classic lost-update and write-skew anomalies. Concurrency control theory offers two

families of solutions [7], [26]. *Pessimistic* control acquires locks before reading data that will be acted upon, so competing transactions wait. *Optimistic* control lets transactions proceed and validates at commit time, aborting on conflict [27]. At the application level, Fowler describes pessimistic and optimistic offline locks and coarse-grained locks that protect an aggregate through its root [14]. In reservation systems the vehicle is a natural lock root: locking the vehicle record before checking for overlapping intervals makes the check-then-insert sequence atomic. On Salesforce, the FOR UPDATE clause locks selected records until the transaction ends; other transactions block for a bounded time, and locks are released if the transaction makes a callout [28]. An alternative is to discretise time into slots and enforce uniqueness on (vehicle, slot), trading flexibility for database-enforced integrity. Interval semantics also matter: using half-open intervals allows back-to-back bookings without false conflicts. Table 1 compares these strategies.

Table -1: Concurrency Strategies for Reservation Integrity

Strategy	Strengths	Weaknesses
Check only (no lock)	Simple; no waits	Race window allows double booking
Pessimistic lock on vehicle	Strong integrity; simple reasoning	Lock waits; released on callouts
Optimistic version check	No waits at read time	Retries under contention; extra field
Slot table + unique key	DB-enforced; very robust	Fixed granularity; many rows
External queue / single writer	Total ordering	Added latency and infrastructure

4.6 Emergency Response and Telematics

Automatic in-vehicle emergency calling became mandatory for new type-approved cars in the European Union through the eCall regulation, which transmits location and vehicle data to the 112 service after a serious crash [8]. Rental fleets without embedded hardware can approximate this with application-level SOS features that capture location through the W3C Geolocation API [29] and open a high-priority incident in the operator's system. The trade-off is reliance on the user's phone, connectivity, and permission prompts. Aftermarket telematics units can add crash detection, odometer and fuel telemetry, and geofencing, but raise cost and privacy concerns.

4.7 Customer Service Recovery and Sentiment

Service failures are inevitable in rental operations, and the quality of recovery strongly affects satisfaction and loyalty [30]. Sentiment analysis research ranges from lexicon methods to supervised and neural classifiers [31]; in practice, explicit star ratings provide a reliable and inexpensive signal. CRM features such as Email-to-Case and SLA milestones turn unstructured complaints into tracked work items, and threshold rules (for example, ratings below three stars) can trigger proactive recovery.

4.8 Governance, Auditability, APIs, and BI

Audit trails that record who changed what and when are fundamental to detecting internal fraud, such as unauthorised discounts or date extensions. Platform features like field history tracking, combined with append-only ledgers for payments, provide this traceability. Secure coding guidance, such as the OWASP Top 10, highlights injection, broken access control, and logging failures as common web risks [32]. REST, as defined by Fielding, remains the dominant style for exposing availability to partners and aggregators [33]. Dimensional modelling principles underpin dashboards for utilisation, revenue, and satisfaction [34], while the DeLone and McLean model offers a framework for evaluating the overall success of such information systems [35].

5. COMPARATIVE ANALYSIS

Table 2 compares the three architectural patterns identified in Sec. 4.1 against the capabilities in the taxonomy. The comparison is qualitative and reflects typical implementations rather than specific products.

Table -2: Comparison of Rental Platform Architectures

Capability	Counter	Web stack	Platform-native
Self-service booking	No	Yes	Yes
Unified customer view	No	Via integration	Native
Voice / chat interface	No	Custom build	Custom build
Reservation integrity	Manual	DB-dependent	Record locks
Digital KYC workflow	No	Custom build	Native records + review
Service case management	Ad hoc	Separate tool	Native
Audit history	Paper	Custom logging	Built-in tracking
Main constraint	Labour, errors	Integration cost	Limits, lock-in

6. REFERENCE CASE: HOMERIDE

Homeride is a platform-native car rental system that serves customers through an Experience Cloud storefront built with Lightning Web Components [36] and operators through a Lightning CRM console. Table 3 maps its eleven modules to the themes of this review.

Table -3: Mapping of Review Themes to the Homeride Reference Case

Theme	Homeride module	Key technique
Cloud & CRM	Four-tier architecture	Trigger handler, Flows, async Apex
Voice & chat	Voice assistant; chatbot	Web Speech API; rule-based NER
Pricing	Cost estimator	Itemised estimate before KYC
Identity	KYC module	Format + checksum; last-4 storage
Concurrency	Overlap engine	Vehicle lock + interval test
Emergency	SOS dispatcher	Geolocation; critical Case
Service	Support; review escalation	Email-to-Case; <3-star rule
Governance & BI	Ledger, audit, dashboards, API	Field history; REST; batch cleanup

Two observations follow from the case. First, the platform-native approach substantially reduces integration effort: SOS events, complaints, and low ratings all become Case records in one queue. Second, several capabilities that appear simple at the interface level depend on careful engineering beneath it. Overlap prevention, for example, requires locking the vehicle before checking intervals; a query alone leaves a race window [28], [7]. Likewise, KYC format validation must be distinguished from proof of identity ownership to remain both effective and lawful [23], [24].

7. OPEN CHALLENGES AND RESEARCH DIRECTIONS

- **Multilingual voice understanding.** Robust slot filling for code-mixed and regional-language speech, ideally with on-device or privacy-preserving processing [16].
- **Lawful, low-friction e-KYC.** Integration of consent-based verification (offline e-KYC, DigiLocker documents) and document image analysis, with DPDP-compliant retention [6], [24].
- **Integrity at scale.** Race-safe reservations across storefront, REST, and bulk channels, including formal verification or property-based testing of overlap logic [26].
- **Connectivity-independent safety.** SMS fallback, offline queuing, and telematics-based automatic crash detection inspired by eCall [8].
- **Explainable dynamic pricing.** Demand-based pricing that remains transparent and fair to customers [19], [5].
- **Empirical evaluation.** Few studies measure booking completion time, abandonment, or incident response in integrated platforms; controlled user studies and field deployments are needed [35], [37].

8. CONCLUSION

Intelligent car rental platforms sit at the intersection of cloud architecture, human–computer interaction, transaction processing, identity law, and service management. This review organised the relevant literature into eight themes, compared architectural and concurrency-control approaches, and illustrated their integration through the Homeride reference case. The analysis shows that platform-native designs reduce integration cost and improve traceability, but that correctness and compliance hinge on details such as record locking, interval semantics, and the distinction between document validation and identity verification. Addressing the open challenges identified here, particularly multilingual voice interaction, lawful e-KYC, and empirical evaluation, will be essential for the next generation of secure and user-centred mobility platforms.

ACKNOWLEDGEMENT

The authors thank Prof. Diksha Kumbhare, Assistant Professor and project guide, and Dr. Pooja Jaiswal, Head of the Department of Computer Science & Engineering – Data Science, Guru Nanak Institute of Technology, Nagpur, for their guidance and support throughout this work.

REFERENCES

1. S. A. Shaheen and A. P. Cohen, "Carsharing and personal vehicle services: Worldwide market developments and emerging trends," *Int. J. Sustain. Transp.*, vol. 7, no. 1, pp. 5–34, 2013.
2. P. Mell and T. Grance, "The NIST definition of cloud computing," NIST, Gaithersburg, MD, USA, Special Publication 800-145, 2011.
3. C. D. Weissman and S. Bobrowski, "The design of the Force.com multitenant internet application development platform," in *Proc. ACM SIGMOD Int. Conf. Management of Data*, 2009, pp. 889–896.
4. E. Adamopoulou and L. Moussiades, "Chatbots: History, technology, and applications," *Mach. Learn. Appl.*, vol. 2, Art. no. 100006, 2020.
5. Baymard Institute, "Cart abandonment rate statistics," Cart & Checkout Usability Research, updated Sep. 2025. [Online]. Available: https://baymard.com/lists/cart-abandonment-rate
6. Unique Identification Authority of India, "Aadhaar paperless offline e-KYC." [Online]. Available: https://uidai.gov.in/
7. P. A. Bernstein and N. Goodman, "Concurrency control in distributed database systems," *ACM Comput. Surv.*, vol. 13, no. 2, pp. 185–221, 1981.
8. European Parliament and Council, "Regulation (EU) 2015/758 concerning type-approval requirements for the deployment of the eCall in-vehicle system based on the 112 service," *Off. J. Eur. Union*, L 123, pp. 77–89, May 2015.
9. B. Kitchenham, "Procedures for performing systematic reviews," Keele Univ., Keele, U.K., Tech. Rep. TR/SE-0401, 2004.
10. Salesforce, "Execution governors and limits," *Apex Developer Guide*. [Online]. Available: https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/apexcode/apex_gov_limits.htm
11. J. -C. Rochet and J. Tirole, "Platform competition in two-sided markets," *J. Eur. Econ. Assoc.*, vol. 1, no. 4, pp. 990–1029, 2003.
12. R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, "Role-based access control models," *Computer*, vol. 29, no. 2, pp. 38–47, Feb. 1996.
13. E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*. Reading, MA, USA: Addison-Wesley, 1994.
14. M. Fowler, *Patterns of Enterprise Application Architecture*. Boston, MA, USA: Addison-Wesley, 2002.
15. D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," *Lingvisticae Investigationes*, vol. 30, no. 1, pp. 3–26, 2007.
16. D. Jurafsky and J. H. Martin, *Speech and Language Processing*, 3rd ed. draft, 2024. [Online]. Available: https://web.stanford.edu/~jurafsky/slp3/
17. W3C Audio Community Group, "Web Speech API," Draft Community Group Report. [Online]. Available: https://webaudio.github.io/web-speech-api/
18. J. Nielsen, *Usability Engineering*. San Diego, CA, USA: Academic Press, 1993.
19. M. K. Geraghty and E. Johnson, "Revenue management saves National Car Rental," *Interfaces*, vol. 27, no. 1, pp. 107–127, 1997.
20. Government of India, "The Motor Vehicles Act, 1988," Act No. 59 of 1988, as amended.
21. J. Verhoeff, *Error Detecting Decimal Codes* (Mathematical Centre Tracts, no. 29). Amsterdam, The Netherlands: Mathematisch Centrum, 1969.
22. Government of India, "The Aadhaar (Targeted Delivery of Financial and Other Subsidies, Benefits and Services) Act, 2016," Act No. 18 of 2016.
23. *Justice K. S. Puttaswamy (Retd.) and Anr. v. Union of India and Ors.*, Writ Petition (Civil) No. 494 of 2012, Supreme Court of India, judgment of Sep. 26, 2018.
24. Government of India, "The Digital Personal Data Protection Act, 2023," Act No. 22 of 2023, Ministry of Law and Justice, Aug. 2023.
25. S. Rose, O. Borchert, S. Mitchell, and S. Connelly, "Zero trust architecture," NIST, Gaithersburg, MD, USA, Special Publication 800-207, Aug. 2020.
26. J. Gray and A. Reuter, *Transaction Processing: Concepts and Techniques*. San Mateo, CA, USA: Morgan Kaufmann, 1993.
27. H. T. Kung and J. T. Robinson, "On optimistic methods for concurrency control," *ACM Trans. Database Syst.*, vol. 6, no. 2, pp. 213–226, 1981.
28. Salesforce, "Locking statements," *Apex Developer Guide*. [Online]. Available: https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/langCon_apex_locking_statements.htm

29. W3C, "Geolocation," W3C Recommendation, 2022. [Online]. Available: https://www.w3.org/TR/geolocation/
30. A. K. Smith, R. N. Bolton, and J. Wagner, "A model of customer satisfaction with service encounters involving failure and recovery," *J. Mark. Res.*, vol. 36, no. 3, pp. 356–372, 1999.
31. B. Pang and L. Lee, "Opinion mining and sentiment analysis," *Found. Trends Inf. Retr.*, vol. 2, no. 1–2, pp. 1–135, 2008.
32. OWASP Foundation, "OWASP Top 10: 2021." [Online]. Available: https://owasp.org/Top10/
33. R. T. Fielding, "Architectural styles and the design of network-based software architectures," Ph.D. dissertation, Dept. Inf. Comput. Sci., Univ. California, Irvine, CA, USA, 2000.
34. R. Kimball and M. Ross, *The Data Warehouse Toolkit*, 3rd ed. Indianapolis, IN, USA: Wiley, 2013.
35. W. H. DeLone and E. R. McLean, "The DeLone and McLean model of information systems success: A ten-year update," *J. Manage. Inf. Syst.*, vol. 19, no. 4, pp. 9–30, 2003.
36. Salesforce, *Lightning Web Components Developer Guide*. [Online]. Available: https://developer.salesforce.com/docs/platform/lwc/guide
37. A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design science in information systems research," *MIS Quart.*, vol. 28, no. 1, pp. 75–105, 2004.