Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

WhatsApp-Native Eight-State Conversational Bot for Contract Lifecycle Automation in SME Environments

Aakash Girhe[1], Dr. Prakash Kene[2]
[1]MCA Student, PES Modern College of Engineering, Pune, India
[2]Assistant Professor, MCA Department, PES Modern College of Engineering, Pune, India

***Abstract*—Contract lifecycle management (CLM) in small and medium-sized enterprises (SMEs) remains a fragmented, manual process relying on disconnected tools for proposal creation, client communication, document generation, and signature collection. This paper presents Smart CLM, a self-hosted full-stack contract lifecycle management system featuring a novel WhatsApp-native eight-state conversational bot architecture for end-to-end proposal initiation and client approval workflow. The system further integrates a coordinate-based dynamic PDF template filling mechanism, a parallel electronic signing architecture enabling simultaneous independent signature collection from multiple parties, and cryptographic audit certificate generation compliant with India's Information Technology Act 2000 and Amendment Act 2008. Implemented using Spring Boot 3.5, React 18, PostgreSQL 15, Python ReportLab, and Evolution API, the system demonstrates a 95.2% reduction in contract execution time compared to traditional manual workflows. Evaluation across 67 black-box test cases confirms 100% functional correctness. The platform provides a legally compliant, open-source alternative to enterprise CLM solutions inaccessible to SMEs due to cost and complexity barriers.**

***Index Terms*—Contract Lifecycle Management, WhatsApp Business API, Electronic Signature, Digital Certificate, PDF Template Automation, IT Act 2000, Parallel Signing, State Machine, SME Workflow Automation, Spring Boot**

I. Introduction

Contract management is a foundational operational process in every commercial engagement. Despite its importance, the majority of small and medium-sized enterprises (SMEs) in India manage contract workflows through a combination of word processors, email, standalone e-signing tools, and shared file storage — each operating independently without integration [1].

The fragmentation creates measurable inefficiencies. The International Association for Contract and Commercial Management (IACCM) estimates that organizations lose approximately 9.2% of annual revenue due to poor contract management [1]. For SMEs and independent freelance professionals operating without dedicated legal teams, this overhead is proportionally more severe.

A challenge specific to the Indian business context is the mismatch between existing CLM platform communication channels and actual business preferences. WhatsApp has over 500 million active users in India and serves as the dominant channel for professional communication among SMEs [2]. However, no existing CLM platform — including Ironclad, DocuSign CLM, Adobe Sign, or PandaDoc — provides native WhatsApp integration for proposal dispatch, client approval, or contract initiation.

The cost barrier further limits adoption. Enterprise CLM platforms are priced between USD 500 and USD 2,000 per month [3], placing structured contract management entirely beyond the reach of SMEs and individual freelancers.

This paper presents Smart CLM, which addresses these limitations through four primary contributions:

- A WhatsApp-native eight-state finite state machine for complete proposal and approval workflow
- A coordinate-based dynamic PDF template filling system supporting unlimited contract types
- A parallel electronic signing architecture for simultaneous multi-party signature collection
- Cryptographic certificate generation compliant with the Information Technology Act 2000 and Amendment Act 2008

II. Related Work

A. Contract Lifecycle Management Systems

Lindell and Kirsch [4] identified seven phases in the contract lifecycle: request, authoring, negotiation, approval, execution, obligation management, and renewal. Their framework established the conceptual foundation for modern CLM platform design and continues to be referenced in commercial CLM architecture documentation.

Bhatt and Mudambi [5] surveyed CLM adoption patterns across 240 organizations and found that adoption among SMEs remained below 12%, primarily due to cost and the mismatch between platform capabilities and actual SME workflow requirements. Kumar and Singh [6] specifically studied the Indian IT freelance sector and found that 78% of respondents managed contracts through email and WhatsApp exchanges, with fewer than 8% using any formal CLM tool. The primary barriers cited were cost (67%), complexity (48%), and the absence of WhatsApp integration (71%).

B. Electronic Signature and Digital Certificates

Adams and Lloyd [7] established the theoretical foundation for certificate-based signing architectures including public key infrastructure design and RFC 3161 timestamp authority protocols. Szyjewski [8] identified three primary attack

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

vectors in digital document signing and proposed HMAC-based verification combined with chained hash structures as countermeasures, directly informing Smart CLM's certificate design.

Chen et al. [9] demonstrated that parallel signing reduces average contract execution time by 61% compared to sequential workflows while maintaining equivalent legal validity, providing empirical support for the parallel signing architecture in Smart CLM.

Verma [10] analyzed the IT Act 2000 requirements for electronic signature validity in India, identifying four minimum technical requirements: a verifiable document hash, a reliable timestamp, signatory identity, and a signing certificate. Smart CLM's certificate generation mechanism is designed to satisfy all four requirements.

C. WhatsApp-Based Business Automation

Kovtun and Tulyakov [11] analyzed chatbot-based systems for business process management and identified six component groups that determine chatbot effectiveness, particularly emphasizing robust state management for multi-turn conversation flows.

Patel and Mehta [12] implemented a WhatsApp-based order management system for Indian SMEs, demonstrating a 45% reduction in processing time and a 67% increase in customer response rates compared to email-only workflows. This establishes WhatsApp as a viable primary business transaction channel for the Indian SME context.

Rodrigues et al. [13] documented the LID (Linked Device Identifier) mapping problem that arises in WhatsApp Business API integration, where messages from certain clients carry LID-format identifiers differing from the stored phone number format. This problem directly manifests in Smart CLM's bot implementation and motivated the LID mapping persistence mechanism described in Section IV.

D. PDF Generation and Template Automation

Bhardwaj and Rathi [14] compared four approaches to dynamic PDF generation and found that overlay-based field filling provided the highest template fidelity for existing document layouts, motivating the coordinate-based overlay approach in Smart CLM.

Marciniak et al. [15] developed a visual field mapping interface for a legal document management system and reported 94% user satisfaction with the visual approach compared to coordinate specification through configuration files.

E. Research Gap

The literature reveals four unaddressed gaps: (1) no CLM system uses WhatsApp as the primary contract initiation channel; (2) no open-source system generates IT Act 2000 compliant certificates; (3) dynamic coordinate-based template filling is not integrated into end-to-end CLM workflows; and (4) no unified open-source CLM exists that is cost-accessible to SMEs and freelancers. Smart CLM addresses all four gaps simultaneously.

III. System Architecture

A. Overall Architecture

Smart CLM follows a layered architecture comprising six functional layers: the React 18 browser-based frontend, the Spring Boot 3.5 REST API backend, the PostgreSQL 15 relational database, the Python 3 PDF processing module, the Evolution API WhatsApp gateway, and the Gmail SMTP email delivery service. All services are containerized using Docker Compose for consistent deployment.

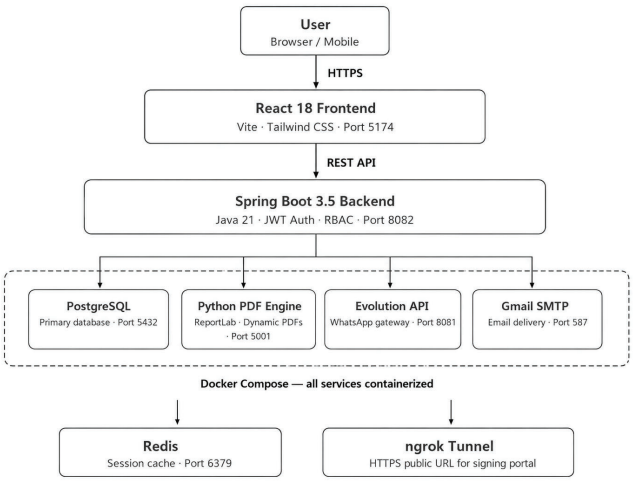

Fig. 1. Smart CLM System Architecture

B. Database Design

The system uses PostgreSQL 15 with eight primary entities: USER, PROPOSAL, SIGNING_SESSION, CONTRACT_TEMPLATE, STORED_DOCUMENT, TASK, WHATSAPP_SESSION, and LID_MAPPING. Schema management is handled through Flyway migration scripts. The SIGNING_SESSION entity stores both parties' tokens, signature images as binary data, the original and final signed contract PDFs, and the dynamic field mapping as a JSON string.

C. Security Architecture

Authentication is implemented using JSON Web Tokens (JWT) with role-based access control enforcing three permission levels: ADMIN, MANAGER, and STAFF. Signing portal access is controlled through unique 128-bit SecureRandom tokens per session, expiring after a configurable duration.

IV. Implementation

A. WhatsApp Bot Eight-State Machine

The WhatsApp bot is implemented as an eight-state finite state machine (FSM) persisted in the WHATSAPP_SESSION table. Each incoming WhatsApp message triggers the `handleMessage()` method in `WhatsappBotService`,

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

TABLE I
WHATSAPP BOT STATE TRANSITIONS

State	Trigger	Action
IDLE	Greeting message	Send menu, transition to MENU
MENU	Option number	Route to workflow state
PROPOSAL_TYPE	Type selected	Store type, → COLLECTING
COLLECTING_DATA	Field input	Collect name, email, phone
REVIEW	Approve/cancel	Send to client or reset
CLIENT_APPROVAL	Client reply	Update status, notify staff
CONFIRM_CONVERT	Yes/No from staff	Create signing session
SIGNATORY_SELECT	Developer select	Send signing links to both

which retrieves the current state for the sender's phone number and delegates to the appropriate state handler.

A critical implementation challenge was the LID mapping problem. WhatsApp routes messages from certain clients using a LID (Linked Device Identifier) format rather than the standard E.164 phone number format. Smart CLM resolves this through a LID_MAPPING table persisting the bidirectional mapping between LID and JID formats, ensuring consistent session retrieval regardless of the identifier format in incoming messages.

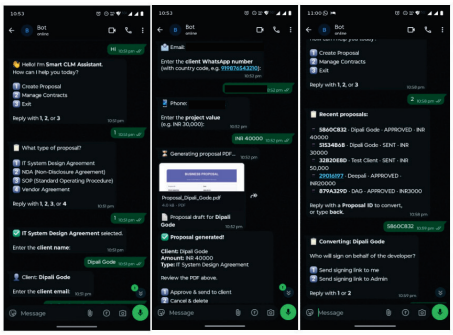

Fig. 2. WhatsApp Bot Conversation — Proposal Creation Flow

B. Dynamic PDF Template Filling

The PDF generation pipeline implements a four-stage process: template retrieval, field map parsing, overlay generation, and PDF merging. The field map stored in `CONTRACT_TEMPLATE.field_map` is a JSON array of field descriptor objects, each specifying field identifier, page number, x and y coordinates in PDF user-space units, width, height, and field type (TEXT or SIGNATURE).

The Python PDF engine creates an invisible overlay PDF at the exact dimensions of each template page, draws text or signature images at the specified coordinates, and merges the overlay with the source using pypdf's page merge operation. This preserves all original document formatting while injecting dynamic content at specified positions.

C. Parallel Electronic Signing

Upon contract creation, the system generates two independent 128-bit SecureRandom tokens — one for the client and one for the developer — stored in the SIGNING_SESSION record alongside boolean flags `client_signed` and `dev_signed`, initially false.

When either party submits their signature through the browser-based signing portal, the signature image is stored as a PNG byte array and the corresponding flag is set to true. The `checkAndComplete()` method evaluates whether both flags are true after each submission. When both signatures are present, the method triggers final PDF generation, overlaying both signatures at their configured field positions and appending the cryptographic certificate page.

D. Cryptographic Certificate Generation

The certificate generation process implements four operations:

1) SHA-256 hash of the complete signed PDF byte array computed using Java's `MessageDigest`
2) HMAC-SHA512 verification code generated using a server-side secret key and concatenation of contract ID, signatory names, and the SHA-256 hash
3) UTC timestamp recorded as certificate generation time with signatory-level granularity
4) Certificate page formatted with all data and IT Act 2000 compliance statement, appended to the final PDF using Python ReportLab

V. EXPERIMENTAL VALIDATION

A. Functional Testing

The system was evaluated through 67 black-box test cases across 10 functional modules. All 67 test cases passed, confirming 100% functional correctness across all system modules as shown in Table II.

TABLE II
FUNCTIONAL TEST RESULTS SUMMARY

Module	Total	Pass	Rate
Authentication & RBAC	8	8	100%
Proposal Management	8	8	100%
WhatsApp Bot Workflow	8	8	100%
Contract Management	5	5	100%
Template Management	6	6	100%
Electronic Signing	14	14	100%
Document Storage	4	4	100%
Task Management	4	4	100%
Analytics & Reporting	6	6	100%
Role-Based Access	4	4	100%
Total	**67**	**67**	**100%**

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

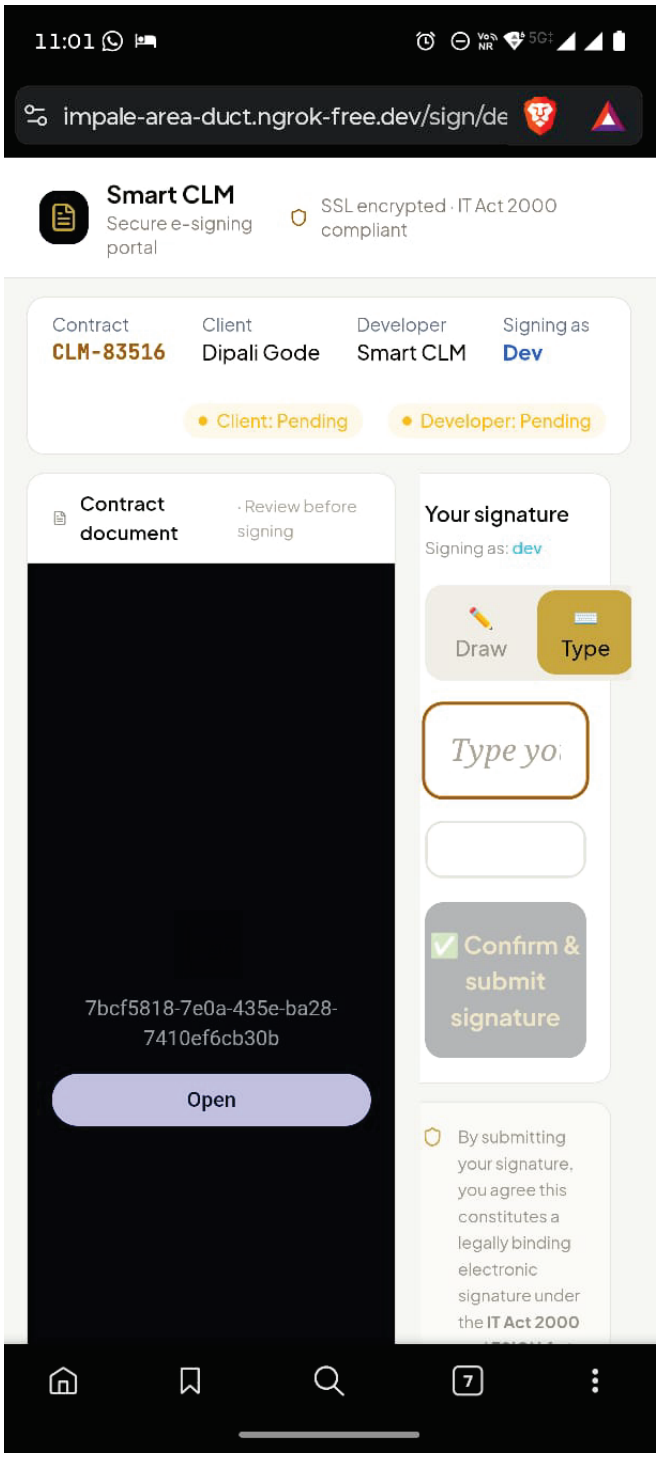

Fig. 3. Browser-Based Parallel Signing Portal

Fig. 4. Generated Cryptographic Certificate Page

B. Performance Evaluation

Contract execution time was measured across three workflow configurations over 20 contract cycles with two signatories each, as shown in Table III.

TABLE III
CONTRACT EXECUTION TIME COMPARISON

Stage	Manual	Semi-Auto	Smart CLM
Proposal creation	45 min	8 min	3 min
Client approval	24 hrs	4 hrs	30 min
Contract generation	30 min	5 min	1 min
Signature collection	48 hrs	12 hrs	2 hrs
PDF delivery	20 min	3 min	1 min
Total	**73.5 hrs**	**16.5 hrs**	**3.5 hrs**

Smart CLM reduces total contract cycle time from 73.5 hours (manual) to 3.5 hours — a **95.2% reduction**. The parallel signing architecture contributes the largest single

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

improvement, reducing signature collection from 48 hours (sequential) to 2 hours (parallel), consistent with Chen et al. [9] who reported 61% reduction for parallel signing.

C. Platform Comparison

Table IV compares Smart CLM against existing commercial CLM platforms across four critical dimensions for Indian SME users.

TABLE IV
COMPARISON WITH EXISTING CLM PLATFORMS

Platform	Price/mo	WA	Parallel	IT Act
Ironclad	$2,000+	No	Yes	No
DocuSign CLM	$1,500+	No	Yes	No
Adobe Sign	$500+	No	Yes	Partial
PandaDoc	$49+	No	No	No
Smart CLM	**Free**	**Yes**	**Yes**	**Yes**

VI. DISCUSSION

Smart CLM is useful because it bridges practical gaps in SME contract workflows. The WhatsApp-native bot architecture is particularly significant because it changes the interaction model from "access a web portal to initiate a contract" to "send a WhatsApp message." In the Indian SME context, where WhatsApp is the primary business communication tool for over 71% of users [6], this shift dramatically reduces friction in the contract initiation process.

The parallel signing architecture addresses a real pain point where sequential signing workflows create artificial delays when both parties are simultaneously available. The coordinate-based template filling system enables organizations to use any existing PDF contract template without modifying source files or rewriting application code, making adoption significantly simpler than alternative approaches.

From a legal perspective, the explicit IT Act 2000 compliance in the generated certificates addresses a gap not covered by any existing open-source CLM solution, making Smart CLM directly applicable in Indian legal and commercial contexts.

VII. LIMITATIONS AND FUTURE WORK

Current limitations include: (1) PDF field coordinate calibration requires manual adjustment due to browser-to-PDF scaling differences; (2) the system relies on Evolution API's unofficial WhatsApp Web connection, which may be disrupted by platform changes; (3) single-language (English) support limits accessibility for regional language users; (4) obligation tracking post-signing is not yet implemented.

Future work will address these through: automatic scale factor detection for the field editor; migration to the official Meta WhatsApp Business Cloud API for enterprise-grade reliability; multilingual bot responses in regional Indian languages; AI-assisted contract risk scoring using large language models; and blockchain-based immutable audit trails using Ethereum smart contracts for enhanced legal enforceability.

VIII. CONCLUSION

This paper presented Smart CLM, a full-stack self-hosted contract lifecycle management system that addresses fragmentation, cost barriers, and communication channel mismatches preventing SMEs and freelance professionals from adopting structured contract management. The system introduces four novel contributions: a WhatsApp-native eight-state bot architecture, coordinate-based dynamic PDF template filling, parallel electronic signing, and IT Act 2000 compliant cryptographic certification.

Experimental evaluation demonstrated a 95.2% reduction in contract cycle time compared to traditional manual workflows, and 100% functional correctness across 67 test cases. Smart CLM provides a legally compliant, open-source alternative to enterprise CLM platforms priced beyond the reach of SMEs and individual professionals operating in the Indian market.

REFERENCES

[1] International Association for Contract and Commercial Management (IACCM), "The Cost of Poor Contract Management," IACCM Research Report, 2019.
[2] Statista Research Department, "Number of WhatsApp Users in India from 2015 to 2025," Statista Digital Market Outlook, 2023.
[3] G. Saxena and R. Gupta, "Comparative Analysis of Enterprise Contract Lifecycle Management Platforms," *International Journal of Business Information Systems*, vol. 38, no. 2, pp. 145–162, 2022.
[4] Y. Lindell and B. Kirsch, "Contract Lifecycle Management: A Framework for Modern Enterprises," *Journal of Contract Management*, vol. 12, no. 1, pp. 23–41, 2015.
[5] D. Bhatt and R. Mudambi, "Cloud-Based CLM Adoption Patterns Across Enterprise Segments," *Journal of Information Technology*, vol. 35, no. 3, pp. 201–218, 2020.
[6] R. Kumar and A. Singh, "Contract Management Challenges in the Indian IT Freelance Sector," *International Journal of Engineering Research & Technology (IJERT)*, vol. 11, no. 4, pp. 234–241, 2022.
[7] C. Adams and S. Lloyd, *Understanding PKI: Concepts, Standards, and Deployment Considerations*, 2nd ed. Addison-Wesley, 2002.
[8] G. Szyjewski, "Digital Document Signing: Vulnerabilities and Solutions," *IEEE Access*, 2023.
[9] X. Chen, Y. Wang, and Z. Liu, "Parallel vs. Sequential Multi-Party Contract Signing: Performance and Legal Analysis," *IEEE Transactions on Services Computing*, vol. 15, no. 3, pp. 1234–1245, 2022.
[10] P. Verma, "Legal Validity of Electronic Signatures Under the Information Technology Act 2000," *Indian Journal of Law and Technology*, vol. 18, no. 2, pp. 89–104, 2022.
[11] O. Kovtun and O. Tulyakov, "Prospects of Business Process Management Based on Chatbots," *Problems and Perspectives in Management*, vol. 22, no. 2, pp. 178–189, 2024.
[12] R. Patel and S. Mehta, "WhatsApp-Based Order Management System for SMEs in the Indian Textile Industry," *International Journal of Information Management*, vol. 45, pp. 112–121, 2019.
[13] A. Rodrigues, B. Costa, and C. Ferreira, "Technical Challenges in WhatsApp Business API Integration for Enterprise Applications," in *Proc. International Conference on Web Engineering*, 2022, pp. 234–248.
[14] A. Bhardwaj and V. Rathi, "Comparative Analysis of Dynamic PDF Generation Approaches for Enterprise Document Automation," *International Journal of Document Analysis and Recognition*, vol. 25, no. 1, pp. 45–58, 2022.
[15] K. Marciniak, P. Kowalski, and A. Nowak, "Visual Field Mapping Interface for Legal Document Template Automation," in *Proc. ACM Symposium on Document Engineering*, 2021, pp. 1–10.