WeRize FinOps Chatbot — Proof of Concept (POC)

---

1. Executive Summary

WeRize is India's first socially distributed full-stack financial services platform, focusing on underserved customers in small towns. They manufacture and distribute lending, banking, group insurance, and savings products via a network of local partners.

Problem: Customer support, onboarding, and financial servicing at scale are limited by human capacity, regulatory requirements, and the need for multilingual support.

Solution: Build a production-ready chatbot with FinOps capabilities to handle common queries (loan status, KYC, product discovery, payments) while ensuring compliance, security, and scalability.

Expected Outcomes: - Reduce support costs and improve response times. - Enable self-service for customers. - Support agents with AI-assisted workflows. - Ensure compliance with Indian NBFC/KYC regulations.

---

2. Business Use Cases

MVP (Phase 1)

1. **Loan Status Lookup**: Balance, EMI schedule, next payment.
2. **KYC Support**: Required documents, onboarding status.
3. **Product Discovery**: Suggest loan/insurance based on profile.
4. **Payments**: Share payment link, confirm last payment.
5. **Escalation**: Seamless ticket creation with context.
6. **Audit Trail & Consent**: Capture user consent for sensitive ops.

Phase 2

- Loan top-up/refinance quotes.
- Insurance claims workflows.
- FinOps guidance (budgeting, savings tips).
- Multi-language support (Hindi + regional).
- Agent-assist (summaries, suggested replies).

Advanced

- RAG (Retrieval Augmented Generation) over policies, SOPs, ledgers.
- Real-time integration with core systems.
- Personalization from user data.
- Regulatory compliance workflows.

3. Architecture Overview

Components: - **Frontend:** Web/mobile chat UI (Gradio/React for POC, React/Vue in prod). - **Backend/API Gateway:** Session mgmt, auth, request routing. - **Chatbot Service:** Controller, NLP model, RAG retriever. - **Data Connectors:** Core banking APIs, CRM/ticketing, logging store. - **Security:** Secrets manager, PII redaction, consent capture. - **Monitoring:** Metrics, cost tracking, audit logs.

Flow: User → Chat UI → API Gateway → Chatbot Controller → {RAG Retriever + Vector DB; Business APIs} → LLM (Claude/OpenAI) → Response → UI

---

4. Data Sources & Ingestion

Initial Data to Index: - Public help pages & policies (werize.com). - Terms & Conditions, fees, product pages. - SOPs (internal underwriting, eligibility rules).

Pipeline: 1. Crawl & clean website docs. 2. Chunk into 512–1024 tokens. 3. Embed with MiniLM or OpenAI embeddings. 4. Store vectors in Pinecone with metadata. 5. Retrieval: Top-K + rerank → inject into prompt.

---

5. LLM & Prompting Strategy

Model Choices: - Anthropic Claude 3 (Haiku for fast/cheap, Opus for quality). - OpenAI GPT-4o for fallback/multilingual.

System Prompt Example:

```
You are WeRize Assistant. Use factual data from retrieved docs, cite sources,
never reveal PII. Confirm identity for transactions. Escalate sensitive cases to
human agents.
```

Prompting Template (RAG):

```
Context:
{retrieved_docs}

User Query: {query}

Answer concisely in ≤100 words, cite sources, and ask clarifying question if
needed.
```

---

6. User Experience Design

• **Identity Verification:** OTP/secret Q before loan status.
• **Consent:** Explicit agreement before data access.
• **UI Features:**
• Quick reply buttons (View EMI, Pay Now, Escalate).
• Typing indicators.
• Escalation handoff with transcript.
• **Multilingual:** Hindi/regional roadmap.

---

7. Security & Compliance

• Use env vars for API keys.
• Encrypt PII at rest and in transit.
• Redact logs; store immutable audit logs.
• Regulatory safety: chatbot never approves loans; only explains pre-approved offers.
• Data retention policy with opt-out.

---

8. Observability & Cost Management

• Track latency, tokens, cost, error rates.
• Example SLOs:
• 95% responses < 2s (non-RAG).
• 99.9% uptime for API.
• Cost controls:
• Limit `max_tokens`.
• Summarize old messages.
• Cache frequent FAQ answers.

---

9. Sample Conversations

Loan Status Check - User: “What’s my outstanding balance?” - Bot: “Verify with OTP sent to +91-xxxx. Proceed?” → verifies → “Outstanding principal ₹X, next EMI DATE. [Pay Now].”

Product Discovery - User: “Can I get a top-up loan?” - Bot: “Based on your profile, you’re pre-qualified for Loan A ₹X–₹Y at Z%. Apply?”

Claims - User: “How do I file insurance claim?” - Bot: “Steps: 1. Collect docs 2. Submit via app. Shall I create claim ticket?”

Escalation - Bot: “I’ll create a support ticket with your transcript. Confirm?”

---

10. Implementation Roadmap

1. Build Gradio chat UI with Claude backend.
2. Index public help pages into Pinecone.
3. Implement RAG retriever + prompt template.
4. Add loan API (read-only) integration.
5. OTP verification before sensitive ops.
6. Add logging, consent, monitoring.
7. Escalation to CRM with context.
8. Pilot with internal users → refine.

---

11. Future Enhancements

- Regional language support.
- Offline/local LLM fallback (Mistral, LLaMA).
- Real-time FinOps insights.
- Integration with analytics for personalized nudges.
- Human agent dashboard for escalations.

---

12. Risks & Mitigations

- **Hallucinations:** Always cite sources, restrict actions.
- **Data Leakage:** Redact logs, consent management.
- **Regulatory:** Consent & immutable logs.
- **Cost Runaway:** Token caps + caching.

---

13. Next Steps

- Build a **starter repo** (Gradio + Claude + Pinecone + sample WeRize data).
- Prepare **security checklist** for Indian fintech compliance.
- Define **audit log schema** (session id, query, response, consent flag).

Prepared for: WeRize FinOps Chatbot POC