Detailed Project Report

Project Title

SupplyChainGPT - A RAG-Powered Generative AI Co-Pilot for Smart Inventory Planning

1. Project Requirements

Objective:

To design and develop an intelligent AI system that acts as a decision-support co-pilot for supply chain and warehouse managers. The system retrieves relevant documents, predicts warehouse demand, and generates actionable recommendations.

Functional Requirements:

- Accept natural language queries.
- Retrieve business documents dynamically.
- Forecast demand and reorder quantities.
- Generate summarized responses with citations.

Non-Functional Requirements:

- Role-based access control.
- Fast response time (<5s).
- Secure data handling.
- Modular extensibility.

2. Project Uniqueness

SupplyChainGPT combines Predictive ML, Generative AI, and RAG into one unified system. It delivers context-aware, explainable, and conversational recommendations instead of static reports. Unlike existing dashboards, it provides dynamic reasoning and auto-cited answers.

3. Why This Project is Necessary

Traditional forecasting systems lack business reasoning. SupplyChainGPT bridges this by:

- Reducing manual data interpretation.
- Integrating forecasts with business context.

- Offering unified knowledge retrieval across silos.
- Providing explainable outputs.
- Enabling faster, informed decisions.

4. Project Flow and Need

Flow:

1. Document ingestion -> parsing and embedding.
2. Data storage in SQL + Chroma.
3. ML forecasting -> SKU-wise predictions.
4. RAG retrieval of relevant content.
5. LLM synthesis -> human-readable output.
6. Streamlit UI -> chat-based visualization.
7. Evaluation -> recall, latency, feedback.

Need:

This flow ensures explainability, modularity, and scalability.

5. Technical Bases to be Covered

Domains and Tools:

- Data Engineering: pandas, pdfminer, openpyxl.
- Embeddings: sentence-transformers, FAISS, Chroma.
- RAG Frameworks: LangChain, LlamaIndex.
- ML Forecasting: Prophet, XGBoost.
- LLM Integration: GPT-4-turbo, transformers.
- Backend: FastAPI.
- Frontend: Streamlit, Plotly.
- Evaluation: Recall@k, MAPE, Faithfulness.

6. GenAI Requirements

LLM: GPT-4-Turbo or Llama 3.

Embeddings: all-MiniLM-L6-v2.

Prompt Design: Structured with context + ML insights.

Vector DB: Chroma -> Pinecone.

Evaluation: Faithfulness, citation accuracy, factuality.

Retrieval: Hybrid semantic + keyword with re-ranking.

7. ML Requirements

Data Inputs:

- SKU-wise demand history.
- Warehouse stock levels.
- Supplier lead times.

Models:

- Demand Forecasting: Prophet/XGBoost.
- Lead-Time Prediction: RandomForest.
- Safety Stock Optimization: Bayesian/Rule-based.

Outputs integrated into RAG as text insights for contextual reasoning.

8. Use of Retrieval-Augmented Generation (RAG)

Purpose: Enhance factual accuracy and context grounding.

Workflow:

1. Query embedding.
2. Retrieval of top-k chunks.
3. Context assembly.
4. LLM synthesis with citations.

Advantages:

- Accurate, verifiable responses.
- Self-updating knowledge base.
- Explainable recommendations.

9. Summary and Expected Results

SupplyChainGPT unifies ML, RAG, and GenAI to make supply chain planning intelligent and explainable.

Expected Results:

- Forecast accuracy ?10% error.

- Retrieval recall ?85%.
- Latency ?5s.
- Hallucination rate <5%.
- User satisfaction ?4.5/5.

Future Scope:
- Multilingual support.
- RL-based stock optimization.
- ERP integrations.
- Voice-based assistant.

Conclusion

SupplyChainGPT modernizes warehouse management through hybrid intelligence - merging data-driven forecasting with generative reasoning. It ensures transparency, speed, and reliability for data-backed inventory decisions.