📦 Smart Inventory Management System

Professional-Grade Inventory Management with AI-Powered Recommendations, built using pure Python (Tkinter), without web framework complexity.

📋 Table of Contents

1️⃣ Introduction

The **Smart Inventory Management System** is a desktop-based application designed to manage products, stock levels, and demand forecasting efficiently. It integrates basic AI logic to provide smart recommendations for restocking and inventory planning. The system is suitable for small businesses, academic projects, and learning purposes.

2️⃣ Quick Start Guide

Step 1: Install Dependencies

```
pip install pandas reportlab
```

Step 2: Run the Application

```
cd C:\Users\Arvindsinh\Desktop\project

python main.py
```

Step 3: Login Credentials

Username: admin

Password: admin123

3️⃣ Project Structure

```
smart-inventory/
|
├── main.py
├── config/
|   └── styles.py
├── ui/
|    ├── app.py
|    ├── header.py
|    ├── sidebar.py
|   └── widgets.py
├── pages/
|    ├── dashboard.py
|    ├── add_product.py
|    ├── update_stock.py
|    ├── ai_insights.py
|    ├── pdf_report.py
|   └── export.py
├── final_smart_inventory_agent.py
├── inventory.csv
└── README.md
```

This modular structure ensures clean separation of UI, logic, configuration, and data.

4️⃣ Features Overview

📊 Dashboard

- Total products count
- Total stock units
- Low stock alerts
- Real-time inventory table

➕ Add Product

- Add new products with validations
- Prevent duplicate product IDs

🔄 Update Stock

- Update existing product stock
- Instant data persistence

🤖 AI Insights

- Risk classification (High, Medium, Safe)
- 7-day demand forecast
- Smart reorder recommendations

📄 PDF Report

- Professional PDF generation
- Timestamped reports

💾 Export CSV

- Export inventory data
- Excel-compatible format

5️⃣ System Architecture

The system follows a layered architecture:

- **Entry Point**: main.py
- **UI Layer**: Tkinter-based UI components
- **Logic Layer**: InventoryAgent (AI & data handling)
- **Data Layer**: CSV file storage

This design improves maintainability and scalability.

6️⃣ File Descriptions

main.py

- Application entry point
- Initializes and launches UI

config/styles.py

- Centralized colors and fonts
- UI consistency management

ui/app.py

- Main application controller
- Handles login and navigation

ui/header.py

- Top header bar
- Displays app title and user

ui/sidebar.py

- Navigation menu
- Page switching logic

ui/widgets.py

- Reusable UI components
- Metric cards, form fields

pages/*.py

- Individual functional pages
- Dashboard, forms, reports

final_smart_inventory_agent.py

- Backend logic
- Inventory operations
- AI recommendation engine

7️⃣ Usage Guide

Adding a Product

1. Open **Add Product** page
2. Enter product details

3. Click **Add Product**
4. Confirmation message displayed

Updating Stock

1. Open **Update Stock** page
2. Select product
3. Enter new stock value
4. Save changes

Viewing AI Insights

1. Open **AI Insights** page
2. Review risk status and suggestions

Generating Reports

- PDF and CSV reports can be generated from respective pages

8️⃣ AI Recommendation Algorithm

Logic Used

weekly_forecast = daily_demand * 7

priority_score = (daily_demand * 2) - current_stock

Decision Rules

- **High Risk**: Stock < low_limit
- **Medium Risk**: Priority score > 0
- **Safe**: Otherwise

This helps in proactive inventory planning.

9️⃣ Color Scheme & UI Design

Purpose	Color
Primary	Dark Blue
Secondary	Sky Blue
Success	Green
Warning	Amber
Danger	Red

Purpose	Color
Background	Light Gray

All colors are centrally managed in styles.py.

🔟 Technologies & Dependencies

Component	Technology
Language	Python 3.x
GUI	Tkinter
Data Handling	Pandas
Reports	ReportLab
Storage	CSV

1️⃣1️⃣ Configuration Options

- Change low stock threshold in final_smart_inventory_agent.py
- Modify login credentials in ui/app.py
- Update CSV file path if required

1️⃣2️⃣ Code Quality Highlights

- Modular and clean design
- Reusable UI components
- Centralized configuration
- Proper error handling
- Readable and maintainable code

1️⃣3️⃣ Future Enhancements

- Database integration (SQLite/PostgreSQL)
- User authentication system
- Charts and analytics
- Mobile application support

- Cloud backup and API integration

1️⃣4️⃣ Troubleshooting

Issue	Solution
App not starting	Check Python installation
Login error	Use admin/admin123
Missing library	Install required package
PDF error	Install reportlab

1️⃣5️⃣ Project Statistics

- Files: 13
- Lines of Code: ~1000
- Dependencies: 2
- Setup Time: < 5 minutes

1️⃣6️⃣ Conclusion

The **Smart Inventory Management System** is a well-structured, AI-assisted desktop application demonstrating practical use of Python, data handling, and UI design. It is suitable for academic submission as well as real-world small inventory use cases.

Last Updated: February 17, 2026