**Project Objective:** The project aims to optimize inventory management for a retail business by leveraging data-driven insights. The goal is to reduce excess inventory, minimize stockouts, and enhance overall supply chain efficiency. This involves analyzing historical sales data, forecasting demand, and recommending inventory levels.

**Design Thinking Process:**

1. **Empathize:** Understand the retail business, its supply chain, and pain points related to inventory management.
2. **Define:** Define the problem and project goals – improving inventory management and supply chain efficiency.
3. **Ideate:** Brainstorm data sources and strategies for data collection, analysis, and forecasting.
4. **Prototype:** Plan the data collection, analysis, and visualization processes.
5. **Test:** Gather and analyze data to create demand forecasts and inventory recommendations.
6. **Implement:** Develop a system to adjust inventory levels based on forecasts.
7. **Iterate:** Continuously refine inventory management strategies based on data insights.

**Development Phases:**

1. **Data Collection:** Gather historical sales data, supplier lead times, and other relevant supply chain data.
2. **Data Preprocessing:** Clean, transform, and prepare the data for analysis.
3. **Database Setup:** Create a database to efficiently store and manage the data, possibly using SQL or NoSQL databases.
4. **Analysis Techniques:** Utilize time series forecasting methods (e.g., ARIMA, exponential smoothing, or machine learning models), demand prediction models, and reorder point calculations.
5. **Visualization Methods:** Create visualizations of demand forecasts, inventory levels, and order recommendations.
6. **Insight Generation:** Identify optimal reorder points, reorder quantities, and stockout risks based on the analysis.
7. **Recommendations:** Develop an automated inventory management system that adjusts inventory levels based on demand forecasts.

**Selected Dataset:** The dataset includes historical sales data, supplier lead times, product information, order history, and inventory levels. It may consist of features such as product ID, date, sales quantity, supplier lead times, and current inventory levels.

**Database Setup:** A relational or NoSQL database is set up to store and manage the dataset efficiently. Tables are created to store product data, sales history, order information, and inventory levels.

**Analysis Techniques:**

1. **Time Series Forecasting:** Use methods like ARIMA or Prophet to forecast future demand based on historical sales data.
2. **Demand Prediction Models:** Develop models to predict future demand based on various factors, including seasonality, promotions, and external events.
3. **Reorder Point Calculation:** Calculate reorder points and reorder quantities based on demand forecasts, lead times, and desired service levels.

**Visualization Methods:**

1. **Demand Forecasts:** Line charts to visualize historical sales and future demand forecasts.
2. **Inventory Levels:** Bar charts or area charts showing current inventory levels and projected inventory levels.
3. **Order Recommendations:** Visualize recommended order quantities and reorder points.

**Explanation Through Programming Code:** Using programming languages like Python, R, or SQL, the insights from the analysis can be translated into actionable code for inventory management. For example, you can develop scripts that automatically adjust inventory levels based on forecasts, calculate reorder quantities, and monitor stockout risks. Here's a simplified example in Python:

# Sample Python code for adjusting inventory levels

def adjust\_inventory(inventory, demand\_forecast, reorder\_point):

if inventory + reorder\_point < demand\_forecast:

order\_quantity = demand\_forecast - (inventory + reorder\_point)

place\_order(order\_quantity) # Function to place orders with suppliers

return True # An order was placed

return False # No order was placed

# Example usage

current\_inventory = 100

forecasted\_demand = 150

desired\_reorder\_point = 20

if adjust\_inventory(current\_inventory, forecasted\_demand, desired\_reorder\_point):

print("Order placed.")

else:

print("No order needed.")