**Week 1: SQL Database Design and Implementation**

**Objective:**  
Design and implement a SQL database for inventory management.

**Tasks:**

1. **Database Design:**
   * **Identify key entities** such as products, suppliers, sales, and inventory levels.
   * **Define relationships and data fields**:
     + Products: Product ID, Name, Category, Unit Price, Supplier ID.
     + Suppliers: Supplier ID, Name, Contact Info, Address.
     + Sales: Sale ID, Product ID, Quantity Sold, Sale Date.
     + Inventory: Product ID, Quantity on Hand, Reorder Level, Stock Date.
   * **Normalize the schema** (ensure it is in 3rd normal form).
   * **Create an ER diagram** to visualize the schema.

**Deliverables:**

* + Finalized SQL database schema.
  + ER diagram for the design.

1. **Database Implementation:**
   * **Set up SQL Server environment** using Microsoft SQL Server.
   * **Write SQL scripts** to create tables based on the schema.
   * **Populate tables** with sample data (mock data for products, suppliers, sales).
   * **Implement constraints:** primary keys, foreign keys, indexing.

**Deliverables:**

* + SQL script for table creation.
  + SQL script for data population.

1. **SQL Queries Development:**
   * **Inventory monitoring queries:** current stock levels, low-stock triggers.
   * **Sales reporting queries:** daily, weekly, monthly breakdowns; sales by product/category.
   * **Supplier performance queries** based on sales volume.
   * **Stored procedures** to automate stock updates or sales reports.

**Deliverables:**

* + Optimized SQL queries.
  + Automated reports through stored procedures.

**Week 2: Data Warehousing and Python Integration**

**Objective:**  
Implement a data warehouse and ETL processes using Python.

**Tasks:**

1. **Data Warehouse Design and Implementation:**
   * **Set up Azure Synapse Analytics** for data warehousing.
   * **Design the schema** (Star Schema or Snowflake Schema):
     + Fact Table: Sales data.
     + Dimension Tables: Products, Suppliers, Time.
   * **Implement the schema** in Azure Synapse Analytics.
   * **Populate warehouse** with sample data.

**Deliverables:**

* + Data warehouse schema.
  + Sample data loaded into the warehouse.

1. **ETL Processes with Python:**
   * **Extract data from SQL Server** using Python (SQLAlchemy or PyODBC).
   * **Transform data** using Pandas (clean data, handle missing values).
   * **Load data into Synapse Analytics** using Python scripts.
   * **Automate the ETL process** using Azure Data Factory or Airflow.
   * (Optional): Add data validation using tools like "Great Expectations" for quality checks.

**Deliverables:**

* + Python scripts for data extraction, transformation, and loading.
  + Automated ETL process.

**Week 3: Forecasting and Azure Integration**

**Objective:**  
Build demand forecasting models and integrate with Azure for analysis.

**Tasks:**

1. **Demand Forecasting Models:**
   * **Collect historical sales and inventory data** from the warehouse.
   * **Implement time series forecasting models** (ARIMA, SARIMA) using Statsmodels or Prophet.
   * **Train and evaluate models** using metrics such as RMSE or MAPE.
   * **Fine-tune models** based on performance.

**Deliverables:**

* + Forecasting models with performance metrics.

1. **Azure Integration:**
   * **Set up Azure Synapse Analytics** for storing forecast results.
   * **Store model outputs** in Azure Data Lake or Blob Storage.
   * **Use Azure Machine Learning (AML)** to track model experiments and monitor performance.

**Deliverables:**

* + Forecast results stored in Azure Synapse or Azure Data Lake.
  + Azure ML setup for tracking and monitoring models.

**Week 4: MLOps and Model Deployment**

**Objective:**  
Deploy the forecasting models and create dashboards for monitoring.

**Tasks:**

1. **MLOps using MLflow:**
   * **Set up MLflow** to track model versions, parameters, and metrics.
   * **Automate CI/CD pipelines** for model retraining and deployment using Azure DevOps.

**Deliverables:**

* + MLflow setup for model tracking and deployment automation.

1. **Deployment as a Web Service or Dashboard:**
   * Option 1: **Deploy models as a REST API** using Flask or FastAPI for querying forecasts.
   * Option 2: **Deploy models using Streamlit or Dash** for building interactive dashboards.
   * **Host on Azure App Services** or **Azure Kubernetes Services** for scalability.

**Deliverables:**

* + Deployed web service or interactive dashboard for visualizing forecasts.

1. **Final Report and Documentation:**
   * **Document the project process** from database design to model deployment.
   * **Create a presentation** showcasing the key components (e.g., SQL database, data warehouse, forecasting models, etc.).
   * **Version control the project** using GitHub or Azure Repos.

**Deliverables:**

* + Final project report.
  + Presentation slides and code repository.