**Assignment: Manage ShopEasy’s Product Inventory with Azure Cosmos DB for NoSQL**

**Problem Statement**

ShopEasy, an e-commerce platform, needs to manage its product inventory to support global operations. The inventory includes details such as product\_id, name, category, price, and stock\_status (true/false). Your task is to design and implement a solution using **Azure Cosmos DB for NoSQL** to store, manage, and query this inventory data efficiently. Ensure the solution is secure, cost-effective, and scalable to handle millions of products across regions. Use the Azure Portal and the Azure Cosmos DB Python SDK to perform CRUD operations (Create, Read, Update, Delete) and apply appropriate security measures. Provide a cost estimate for a small-scale deployment suitable for ShopEasy’s initial launch.

**Objectives**

* Create an Azure Cosmos DB for NoSQL account, database, and container tailored for product inventory.
* Implement CRUD operations using Python to manage product data.
* Secure the database with appropriate authentication methods.
* Analyze and justify the cost and scalability of your solution.

**Requirements**

* **Data Structure**: Each product should include at least product\_id, name, category, price, and stock\_status.
* **Tools**: Azure Portal, Python 3.7+ with azure-cosmos SDK, Azure CLI (optional), Azure Price Calculator.
* **Security**: Implement access control using Microsoft Entra ID or access keys.
* **Deliverables**:
  + Documentation of your Cosmos DB setup (e.g., account, database, container details).
  + Python script output or code demonstrating CRUD operations.
  + Security configuration details.
  + Cost estimate and rationale for use case.

**Constraints**

* Use **Azure Cosmos DB for NoSQL** (formerly SQL API).
* Minimum throughput: 400 RUs.
* Partition key must optimize query performance for product categories.
* If Azure access is restricted, provide a JSON simulation of your solution.

**Submission**

* Submit a document (text, markdown, or PDF) with:
  1. Cosmos DB setup details (e.g., account name, database, container, partition key).
  2. Python script or JSON simulation of CRUD operations.
  3. Security configuration (e.g., Entra ID role or key usage).
  4. Cost estimate and justification for ShopEasy’s inventory management.
* Include a query to find products by category or stock status.