MLA// OPTIONAL PORTFOLIO TASK

Please make a copy of this document

NAME OF SESSION	Architecture
NAME OF STUDENT	Funkeyi Jessica Omoro

ASSESSMENT FOCUS	Designing a System for a Retail Store

Scenario:

You are a software architect working for a retail company planning to modernize its in-store systems. The company wants to move away from its traditional point-of-sale (POS) terminals and adopt a more flexible and scalable architecture. The new system should support various functionalities, including product lookup, inventory management, customer relationship management (CRM), sales processing, payment handling, real-time analytics, reporting, and integration with mobile devices (for price checks and mobile checkout).

Task:

Design a software architecture for the new in-store system. Focus on these key decisions:

- **Architecture Pattern:** Choose between a monolithic or microservices architecture and briefly justify your choice.
- **Components:** Identify the key components of the system (e.g., product catalog, inventory, payment processing).
- **Communication:** Suggest a suitable communication pattern for component interaction (e.g., REST, SOAP, RCP, GraphQL or message brokers).
- **Data:** Briefly describe how data will be managed in the system.

Deliverables:

- A short written response (approximately 250 words) and a simple diagram to illustrate your proposed architecture choices.

YOUR ANSWER:

Architecture Pattern: I recommend a microservices architecture over monolithic for this retail modernization project. The diverse functionality requirements (POS, inventory, CRM, analytics, mobile integration) naturally align with microservices boundaries, enabling independent scaling, team

autonomy, and technology flexibility.

Key Components: The system consists of eight core microservices: Product Catalogue (managing SKUs and pricing), Inventory Management (real-time stock tracking), Sales Processing (transaction handling), Payment Service (secure payment processing), Customer Service/CRM (profiles and loyalty), Analytics Service (real-time BI), Reporting Service (dashboard aggregation), and Notification Service (multi-channel communications).

Communication: I propose a hybrid approach using REST APIs for synchronous real-time operations, GraphQL for mobile applications requiring flexible data fetching, and Apache Kafka message bus for asynchronous event-driven workflows. An API Gateway provides unified authentication, rate limiting, and routing.

Data Management: Following the "database per service" pattern, each microservice owns its data domain. PostgreSQL handles transactional data (sales, customers), MongoDB manages the flexible product catalogue schema, Redis provides caching for high-frequency operations, and a centralized data lake aggregates analytics. Event sourcing ensures complete audit trails for financial transactions.

This architecture supports the retail environment's demands for high availability, scalability, and integration flexibility while enabling gradual migration from legacy systems.