Final Project Output: Integrated Messaging and XML Data Processing System

Project Title:

"Asynchronous Messaging System with XML Data Transformation"

1. Project Overview

The final project requires students to design, develop, and implement a system that integrates:
✅ **Asynchronous Messaging System** (using RabbitMQ, Apache Kafka, or ActiveMQ)
✅ **XML Data Parsing and Transformation** (using SAX/DOM for parsing and XSL/XSLT for data transformation)
✅ **Operating System Scripting** (to automate deployment, monitoring, or data backup)

2. Key Features and Deliverables

Phase 1: Messaging System Setup

✅ Install and configure a messaging service.
✅ Implement a **Publisher-Subscriber Model** that enables asynchronous message exchange.
✅ Include **error handling** and **message retry mechanisms**.

Deliverable: Working messaging system with a functioning publisher and subscriber.

Phase 2: XML Data Parsing and Transformation

✅ Develop a Java program that uses **SAX** or **DOM** to parse XML data.
✅ Implement **XSL/XSLT** to convert XML data into HTML, JSON, or CSV format.
✅ The system should display or store the transformed data appropriately.

Deliverable: XML parser with successful data transformation.

Phase 3: Operating System Scripting

✅ Develop a **Bash**, **PowerShell**, or **Python** script that performs one of the following:

- Automates deployment of the messaging system.
- Automates file backup or data archiving.
- Monitors the messaging queue for errors or failed messages.

Deliverable: Fully functional script that automates a chosen system task.

Phase 4: System Integration and Testing

✅ Integrate the messaging system, XML parser, and OS automation script into one cohesive system.
✅ Implement **logging** and **error reporting** for tracking system performance.
✅ Conduct unit and integration testing to validate system functionality.

Deliverable: Complete integrated system with test results.

3. Technical Requirements

✅ Programming Language: **Java** (for core system implementation)
✅ Messaging Platform: **RabbitMQ**, **Kafka**, or **ActiveMQ**
✅ XML Parsing Tools: **SAX**, **DOM**, **XSL/XSLT**
✅ Operating System Scripting: **Bash**, **PowerShell**, or **Python**

4. Final Submission Requirements

📂 Source Code Folder Structure

pgsql

Copy code

```
/Final_Project
 ├── /MessagingSystem
 │   ├── Publisher.java
 │   ├── Subscriber.java
 │   ├── queue_config.yaml
 │
 ├── /XMLProcessing
 │   ├── XMLParser.java
 │   ├── data.xml
 │   ├── transform.xsl
 │
 ├── /Scripts
 │   ├── deploy.sh
 │   ├── backup_script.py
 │
 ├── README.md
 ├── ProjectReport.pdf
```

📄 Final Report Guidelines

The report should include the following:

1. **Title Page**
2. **Abstract** (Summary of your project goals and implementation)
3. **Introduction** (Describe the system's purpose and objectives)
4. **System Design** (Detailed architecture, flowcharts, and diagrams)
5. **Implementation Details** (Explain the coding process, tools used, and logic behind each component)
6. **Testing Procedures** (Unit testing, integration testing, and results)

7. **Challenges and Solutions** (Discuss problems faced and how they were resolved)
8. **Conclusion and Recommendations** (Summary of project outcomes and potential improvements)
9. **Appendix** (Source code snippets, configurations, and sample data files)

5. Grading Rubric (100 Points)

Criteria	Excellent (20 pts)	Good (15 pts)	Fair (10 pts)	Needs Improvement (5 pts)
Messaging System Implementation	Fully functional publisher-subscriber model with retries and error handling.	Correct messaging system but lacks advanced features.	Basic messaging system with minor issues.	Incomplete or non-working messaging system.
XML Parsing and Transformation	Accurate XML parsing with XSL/XSLT transformation implemented successfully.	Correct XML parsing but minor transformation issues.	Basic XML parsing with limited transformation features.	No XML parsing or transformation attempted.
OS Scripting Automation	Fully functional automation script with detailed documentation.	Working script with minor functionality gaps.	Script attempted but contains major issues.	No automation script provided.
System Integration & Testing	All components integrated seamlessly with clear documentation and evidence of testing.	Integration mostly functional but missing minor details.	Integration attempted but lacks stability.	No integration attempted.
Final Report Documentation	Comprehensive, detailed report with clear explanations and visuals.	Good documentation but missing some details.	Basic report with minimal explanation.	Incomplete or missing documentation.

📅 **Deadline: Final project submission is due by Week 8.**