AI Prompt Log – Serverless Stock Portfolio Tracker Project

🔹 Prompt 1: Project Architecture Design

“Design a serverless architecture for a real-time stock portfolio tracking system using AWS services. The system should include API Gateway, AWS Lambda, DynamoDB, and an external stock price API. Explain how data flows between services.”

🔹 Prompt 2: AWS Services Selection

“Suggest appropriate AWS services for building a scalable and cost-efficient stock portfolio tracker and justify why each service is suitable for this use case.”

🔹 Prompt 3: DynamoDB Data Modeling

“Design a DynamoDB table schema for storing user portfolios and stock holdings. Include partition keys, sort keys, and access patterns.”

🔹 Prompt 4: Backend API Design

“Create REST API endpoints for managing stock portfolios, including creating portfolios, adding holdings, deleting holdings, and fetching portfolio data.”

🔹 Prompt 5: Profit and Loss Calculation Logic

“Explain how market value, unrealized profit/loss, and profit/loss percentage can be calculated for stock holdings using real-time market prices.”

🔹 Prompt 6: External API Integration

“Explain how an external stock price API (such as Finnhub) can be integrated into an AWS Lambda function to fetch real-time stock prices.”

◆ **Prompt 7: Error Handling Strategy**

“Suggest error handling mechanisms for API Gateway and Lambda functions to manage invalid stock symbols, missing API keys, and external API failures.”

◆ **Prompt 8: CORS and Frontend Integration**

“Explain how CORS should be configured in API Gateway to allow secure communication between a frontend application and backend Lambda APIs.”

◆ **Prompt 9: IAM and Security (Learner Lab Friendly)**

“Suggest a secure IAM strategy for AWS Lambda in a learner lab environment, minimizing permissions while allowing access to DynamoDB and CloudWatch Logs.”

◆ **Prompt 10: SDG Alignment**

“Identify which UN Sustainable Development Goals (SDGs) are aligned with a real-time stock portfolio tracking system and justify the alignment.”

◆ **Prompt 11: Scalability and Performance**

“Explain how a serverless architecture ensures scalability and high availability for a real-time financial application.”

◆ **Prompt 12: Monitoring and Logging**

“Explain how AWS CloudWatch and X-Ray can be used to monitor Lambda execution, performance, and errors in a serverless application.”

◆ **Prompt 13: Project Workflow Explanation**

“Explain the complete workflow of a serverless stock portfolio tracker from frontend request to backend processing and database interaction.”

- **Prompt 14: Limitations and Future Enhancements**

"Identify limitations of the current stock portfolio tracking system and suggest possible future improvements."

- **Prompt 15: Project Poster Design**

"Create a concise academic poster write-up for a serverless real-time stock portfolio tracker. The poster should include the problem statement, system architecture, technologies used, workflow diagram, key features, and alignment with Sustainable Development Goals (SDGs)."