**AUTOMATION USING AZURE DEVOPS**

**Introduction to Azure DevOps**

Azure DevOps is a cloud-based platform that provides tools for project planning, version control, build automation, testing, and deployment. It integrates all stages of the software and data development lifecycle into one platform.   
 In the Retail Sales Performance Dashboard project, Azure DevOps was used to automate data analysis workflows, ETL processes, and report generation—ensuring the system runs automatically every week with minimal manual intervention.

**Need for Automation**

Manual execution of scripts for large retail datasets can be time-consuming and prone to errors. Automation enables consistent and timely execution of all processes—from data collection to reporting.   
 By using Azure DevOps, the entire workflow (Python, PySpark, and Databricks tasks) was automated to generate weekly insights such as revenue, profit margins, and underperforming stores, improving efficiency and accuracy.

**Overview of CI/CD Concepts**

**Continuous Integration (CI)** involves merging updated code into a shared repository, where it is automatically built and tested.   
 **Continuous Deployment (CD)** extends this by automatically executing and deploying the tested code to production or analytics environments.   
 In this project, CI/CD ensured that updates in the code or data triggered an automated pipeline run to refresh the retail performance insights.

**Week 5 Tasks Overview**

During Week 5, the main focus was automating the data pipeline using Azure DevOps. The following key tasks were performed:

* + Environment Configuration
  + YAML Pipeline Creation
  + CI/CD Automation Setup
  + ETL and Report Execution
  + Output File Generation
  + Log Creation and Validation
  + Notification Configuration

**Pipeline Automation Steps**

**Set up the Project Repository:**   
 Create a new project in Azure DevOps and upload all Python, PySpark, and Databricks notebooks.

**Create YAML Pipeline:**   
 Define pipeline stages for environment setup, script execution, and report generation.

**Configure Triggers:**   
 Schedule the pipeline to run weekly or on every data/code update.

**Run Analysis and ETL:**   
 Execute scripts for calculating revenue, profit margins, and identifying underperforming stores.

**Generate and Store Output:**   
 Save the final sales reports as CSV files and push them to shared storage.

**Add Logging and Notifications:**   
 Log the summary metrics and configure email notifications for pipeline success or failure.

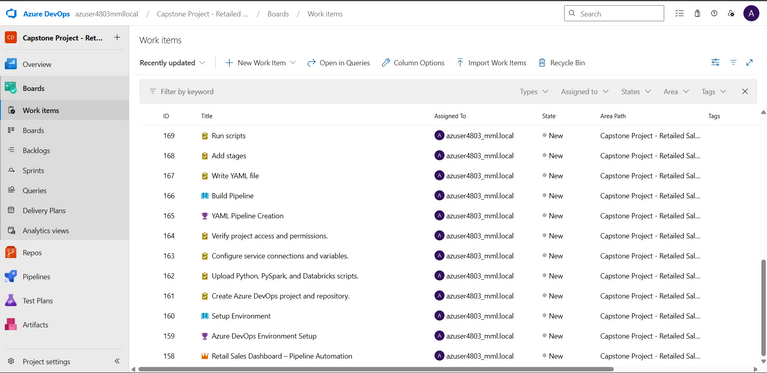
**Expected Outputs and Benefits**

* 1. The Azure DevOps pipeline automated the complete analysis workflow, producing:
  2. Weekly updated sales performance reports.
  3. Logs containing store-level revenue and profit summaries.
  4. Automatic alerts for pipeline completion or errors.

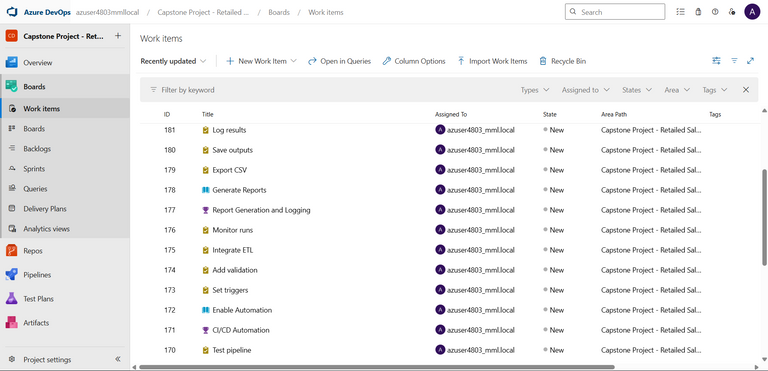
**Benefits:**

* 1. Reduced manual work and execution time.
  2. Consistent and accurate results.
  3. Continuous integration and delivery of updates.
  4. Improved data visibility and business decision-making.

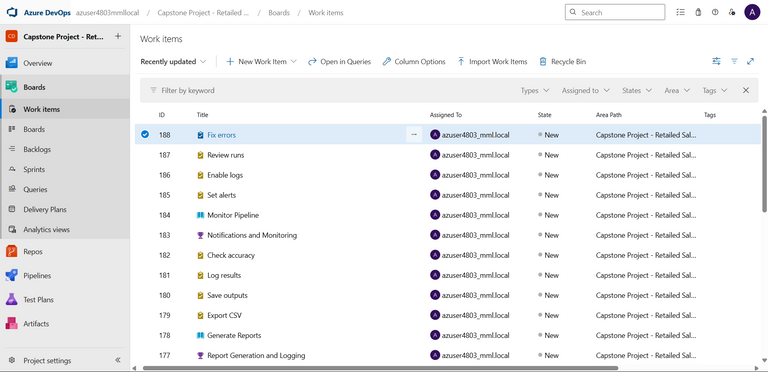
**Screenshots**



*Figure 1: Azure DevOps Board showing initial work items created for automation tasks.*



*Figure 2: Azure DevOps work items for Week 5 – Pipeline Automation phase.*



*Figure 3: Azure DevOps work items for Week 5 – Pipeline Automation phase.*

**Conclusion**

Through Azure DevOps automation, the Retail Sales Performance Dashboard project achieved complete CI/CD integration. Every process—from data extraction and transformation to analytics and reporting—was streamlined and automated.   
 This final automation phase ensured that the dashboard remains continuously updated, reliable, and scalable for future data growth.