# Performance Report

Name: Muhammad-Muneeb-Ur-Rehman

Designation: MTO – MIS Department

Company: Fauji Cement Company Limited

Duration: November 2024 – Present

## 1. Introduction

This report summarizes my professional performance during my tenure at Fauji Cement Company Limited in the MIS Department. It outlines the key contributions I have made to the organization and the goals I aim to achieve in the coming months.

## 2. Work Summary

We previously encountered several challenges that required innovative problem-solving approaches. To address these issues effectively, we developed and implemented a range of strategic solutions. **a. Windows with Pre-Installed Software and Configurations**  
Several computer systems in the department were old and operated at a slower speed, resulting in prolonged installation times for the operating system and essential work-related software. Preparing a single system typically required more than 4 to 5 hours.

To address this issue, I developed a **WinINI.iso** file that automates the installation process. This solution allows the complete setup including the operating system, essential software (such as SAP, EMS, and Microsoft Office), and necessary configurations to be completed within **30 to 40 minutes**, even on older machines.

**b. Automation Scripts**

1. **SAP Logon configuration**  
   The script automates SAP Logon configuration by creating required folders and adding a predefined connection (“FCCL SAP PRODUCTION”) directly to SAPUILandscape.xml.  
   It eliminates manual setup errors and reduces configuration time from 10–15 minutes to under a minute.  
   Built with Python’s standard libraries (xml.etree.ElementTree, os, uuid), it ensures consistent SAP access across all systems.  
   The script verifies and generates necessary XML nodes, router entries, and workspace links automatically.  
   It standardizes SAP GUI setups, improving reliability and efficiency in system deployment.
2. **vnc\_auto\_scan.py – VNC Auto-Discovery & Shortcut Generator**  
   This script scans the network 192.168.1.0/24 to find active VNC servers on port **5900**.  
   It pings each host, checks if the port is open, retrieves hostname, FQDN, and NetBIOS username, then generates .vnc shortcut files with prefilled credentials.  
   The goal is to eliminate the manual process of identifying and configuring VNC connections across multiple systems.  
   It uses **Python’s standard libraries**: socket, ipaddress, subprocess, and concurrent.futures for multithreading.  
   Output files are stored in the vnc\_shortcuts folder, reducing setup time from hours to minutes.  
   Before running, update the network, port, and password in the constants.  
   Run with python vnc\_auto\_scan.py.  
   This improves efficiency, ensures consistency, and simplifies remote access setup.
3. **ping\_test\_utility.py**   
   This script pings a fixed list of critical hosts (3 attempts each), parses RTTs, packet loss, and reachability, then saves a styled Excel report on the Desktop.  
   It solves slow, manual health checks by producing a one-click, color-coded (green/yellow/red) availability dashboard with min/max/avg times.  
   Works on Windows/Linux/macOS: adapts to ping output formats and computes stats even if summaries aren’t present.  
   Excel output (ping\_results.xlsx) includes headers, borders, alignment, conditional fills, and a fallback filename if the file is open.  
   Console shows per-host results plus a tabular summary.  
   Local IP is detected automatically (UDP socket to 8.8.8.8).  
   Tech: Python 3, subprocess, re, platform, socket, openpyxl.  
   Usage: update the hosts list in main(), then run python script.py; open the generated Excel.
4. **Time Machines Status**  
   Pings a fixed list of “Time/Attendance & Solar” devices (3 attempts each) and computes reachability, packet loss, and RTT min/max/avg.  
   Outputs a color-coded Excel report on the Desktop as **time\_machines\_status.xlsx** (styled headers, borders, and per-cell fills).  
   Works on Windows/macOS/Linux by adapting to platform-specific ping formats; still computes stats if summaries are missing.  
   Console shows live per-host results and a final summary table; local IP is auto-detected via a UDP socket to 8.8.8.8.  
   Solves manual, ad-hoc checks by producing a one-click, standardized health dashboard for critical edge devices.  
   Tech: Python 3, subprocess, re, platform, socket, openpyxl, datetime, pathlib.  
   Customize targets in hosts and run: python time\_machines\_status.py.  
   If the Excel file is open, it auto-saves with a timestamped filename.
5. **Enhanced Network Scanner**   
   Scans any CIDR subnet, pings each host, and classifies status: **Active / Host Unreachable / Request Timeout**.  
   For **active** hosts, it resolves DNS hostname, tries NetBIOS computer name, and attempts to fetch the **logged-in username** (WMI/nbtstat/query/PowerShell on Windows).  
   Also tests each active host’s connectivity to defined **critical IPs** (e.g., EMS, SAP), capturing reachability and response time.  
   Multithreaded with progress updates; robust parsing for Windows/Linux/macOS ping outputs.  
   Exports a styled Excel to Desktop: host info plus one column per critical IP (✓ Yes with RTT / ✗ No / ⏱ Timeout), and a summary block.  
   Solves manual, fragmented checks by producing a **single, comprehensive** network health + dependency matrix.  
   Tech: Python 3 stdlib (ipaddress, subprocess, concurrent.futures, socket, re, pathlib) + openpyxl.  
   Run: python enhanced\_network\_scanner.py → enter CIDR (e.g., 192.168.1.0/24).
6. **Network Scanner**   
   Scans any CIDR subnet, pings each host, and classifies status: **Active / Host Unreachable / Request Timeout**.  
   For active hosts, resolves **DNS hostname**, attempts **NetBIOS** computer name, and fetches **logged-in username** (Windows methods).  
   Multithreaded for speed with progress updates; robust parsing for Windows/macOS/Linux ping outputs.  
   Exports a styled Excel (**comprehensive\_network\_scan.xlsx**) to Desktop with color-coded status and a summary block.  
   Solves manual inventory and ad-hoc checks by producing a repeatable host catalog with identity details.  
   Tech: Python stdlib (ipaddress, subprocess, concurrent.futures, socket, re, os, pathlib, time, sys) + openpyxl.  
   Usage: python enhanced\_network\_scanner\_hostname.py → enter CIDR (e.g., 192.168.1.0/24).  
   Note: username/computer name retrieval may require permissions and NetBIOS/WMI availability.

## 3. Skills and Competencies Demonstrated

[List key technical and soft skills you’ve applied — e.g., data analysis, problem-solving, communication, etc.]

## 4. Challenges and How They Were Addressed

[Briefly describe any challenges you encountered and how you resolved or managed them.]

## 5. Future Goals and Development Plan

[Write what you aim to achieve in the next review period — e.g., automation goals, leadership roles, upskilling, or project improvements.]

## 6. Conclusion

In conclusion, my tenure at Fauji Cement Company Limited has been a valuable learning experience. I have contributed positively to the department’s objectives and am committed to achieving higher performance standards in the future.