Institute Management System using CLI

**Submitted By**

Project Leader Name: **Mudasir**  
Project Leader Roll Number: **22K-8732**

**Group Members**

Student-1 Name: **Aman Ullah Kazi**Student-1 Roll Number: **22K-8714**

Student-2 Name: **Mudasir**  
Student-2 Roll Number: **22K-8732**

Student-3 Name: **Arooba Faisal**Student-3 Roll Number: **22K-4083**

Student-4 Name: **Syed Areeb Hussain**Student-4 Roll Number: **22K-4042**

***Department of Computer Science  
National University of Computer & Emerging Sciences***

# 1. **Project Objective**

The objective of this project is to build a **command-line-based Institute Management System using Bash Shell scripting.** The system allows management of students, teachers, courses, enrollments, and grading in a simulated multi-user environment. The project is designed to implement and demonstrate key Operating System (OS) concepts through practical scripting tasks.

**2. Implementation Explanation**

The project is implemented entirely in a .sh Bash script. It includes modular functions for creating and managing entities like semesters, students, teachers, and courses. The script is menu-driven and supports role-based access for three types of users:

* **Admin:** Full permissions (create/view/modify/delete teachers and students both)
* **Teacher:** Limited to viewing enrollments and updating marks
* **Student:** View their own course and performance details

Data is stored and managed using .csv files (acting as flat-file databases). Each operation (e.g., course creation, enrollment, grading) is handled through functions with appropriate checks and balances.

# 3. OS Concepts Implementation

## 3.1 File Handling

Used for persistent data storage through .csv files. Operations include reading, writing, appending, and modifying performing on run time modification in files. The files includes student.csv, teacher.csv, course.csv, courseEnroll.csv, etc.

## **3.2 Process Control and Execution**

Utilizes while loops and case statements to control the flow of execution. Menu-driven interface mimics process scheduling and job control. Functions act like sub-processes to execute specific tasks modularly.

## 3.3 User Management & Access Control

Role-based access is enforced:

* Admin (password protected)
* Teacher (access through teacher ID)
* Student (access through student ID)

This simulates OS-level user privilege management.

## 3.4 Project Category

## Input validation is done using conditional logic (if statements). File existence checks ([ ! -f filename ]) prevent script crashes. Prevents duplicate records and out-of-bound input (e.g. marks exceeding max limit).

## 3.5 Inter-Process Communication (IPC)

## Function return values mimic shared memory between processes. Shared .csv files are accessed and modified by multiple functions.

## 3.6 Scheduling Logic (Simulated via Menus)

## Menu selection logic simulates job/task scheduling in a system. Each role executes specific scheduled tasks based on input.

## 3.7 Grading Logic and Resource Allocation

Students are graded based on attendance, quiz, mid, and final marks. Grade assignment logic simulates how OS assigns resources or priorities.

# 4. Hardware and Software Requirements

## Hardware Requirements

* Computer or laptop for using Ubunto or Linux.

## Software Requirements

* Shell Scripting should be installed in your Ubunto and should also be executable in your terminal.
* Ubunto or Linux (Ubunto highly preferred)
* CSV file reader to overview the run time changes in CSV files.
* No any IDE needed because it is a CLI based project so no need for any GUI as well.

# 5.Conclusion

**The Institute Management System** project has effectively demonstrated the practical application of core Operating System concepts using Bash scripting. It provided hands-on experience in file handling, user-role management, process execution, error control, and data integrity. Through this project, we learned how fundamental OS mechanisms can be simulated and managed within a shell script, making it a valuable and educational experience for understanding how operating systems work behind the scenes.