# Mini Expert System Report - University Logic Rules System

## Introduction

This report presents a mini expert system that implements logical reasoning using implication rules (P → Q) to validate various university policies and procedures. The system tests five different rules and logs all results to a CSV file for analysis.

## Rules Tested

### 1. Attendance Rule

- Logic: If a student is late (P), then they must provide an excuse letter (Q)
- Implementation: P → Q where P = "student is late" and Q = "excuse letter provided"
- Purpose: Ensures students follow proper procedures when arriving late to class
- Outcome: Rule is satisfied when students either arrive on time or provide proper documentation for tardiness

### 2. Grading Rule

Logic: If a student's grade is ≥75 (P), then the student passes (Q)
- Implementation: P → Q where both P and Q evaluate the same condition (grade ≥ 75)
- Purpose: Validates the university's passing grade requirement
- Outcome: Demonstrates the basic pass/fail logic based on numerical grades

### 3. Login System Rule

Logic: If the password is correct (P), then access is granted (Q)
- Implementation: P → Q where P = "password matches 'admin123'" and Q = "access granted"
- Purpose: Simulates a basic authentication system
- Outcome: Access control based on credential verification

### 4. Bonus Points Rule

Logic: If a student has regular attendance (P), then they are eligible for bonus points (Q)
- Implementation: P → Q where P = "regular attendance" and Q = "bonus eligible"
- Purpose: Rewards students for consistent attendance
- Outcome: Incentivizes good attendance behavior through bonus point eligibility

### 5. Library Borrowing Rule (New Rule)

Logic: If a student has a valid ID AND all library fees are paid (P), then they are allowed to borrow books (Q)
- Implementation: P → Q where P = "valid_id AND fees_paid" and Q = "allowed to borrow"
- Purpose: Ensures library resources are available only to eligible students
- Outcome: Combines multiple conditions to determine borrowing privileges

Name: Garin, Jeremy M.
Section: BSCS – 3A

CSST 101
Mr. Bernandino

**Description of the New Library Rule**

The Library Borrowing Rule was added to demonstrate a more complex logical condition that requires multiple prerequisites. This rule implements the following logic:

**Rule Definition:** Students can borrow library books only if they have both a valid student ID and have paid all outstanding library fees.

**Technical Implementation:**
- Uses compound boolean logic: `P = valid_id AND fees_paid`
- Applies implication: If (valid ID AND fees paid) → then (allowed to borrow)
- Both conditions must be true for the rule to grant borrowing privileges

**Real-world Application:**
This rule reflects common library policies where:
1. Valid identification ensures only enrolled students access resources
2. Paid fees ensure students are in good standing with the library
3. Both conditions together maintain library security and financial accountability

**System Behavior:**
- If both conditions are met: "Allowed to borrow ✓"
- If either condition fails: "Not allowed to borrow ✗"
- All results are logged with timestamp and student name for administrative tracking

**System Features**

**Data Management**
- CSV Logging: All rule evaluations are automatically logged with timestamps
- File Organization: Results are moved to a dedicated `data/` folder upon exit
- Duplicate Prevention: Timestamped filenames prevent data loss when multiple sessions occur

**User Interface**
- Menu-driven System: Clear navigation through numbered options
- Input Validation: Handles various input formats and provides error feedback
- Result Display: Shows logical variables and outcomes in readable format

**Technical Architecture**

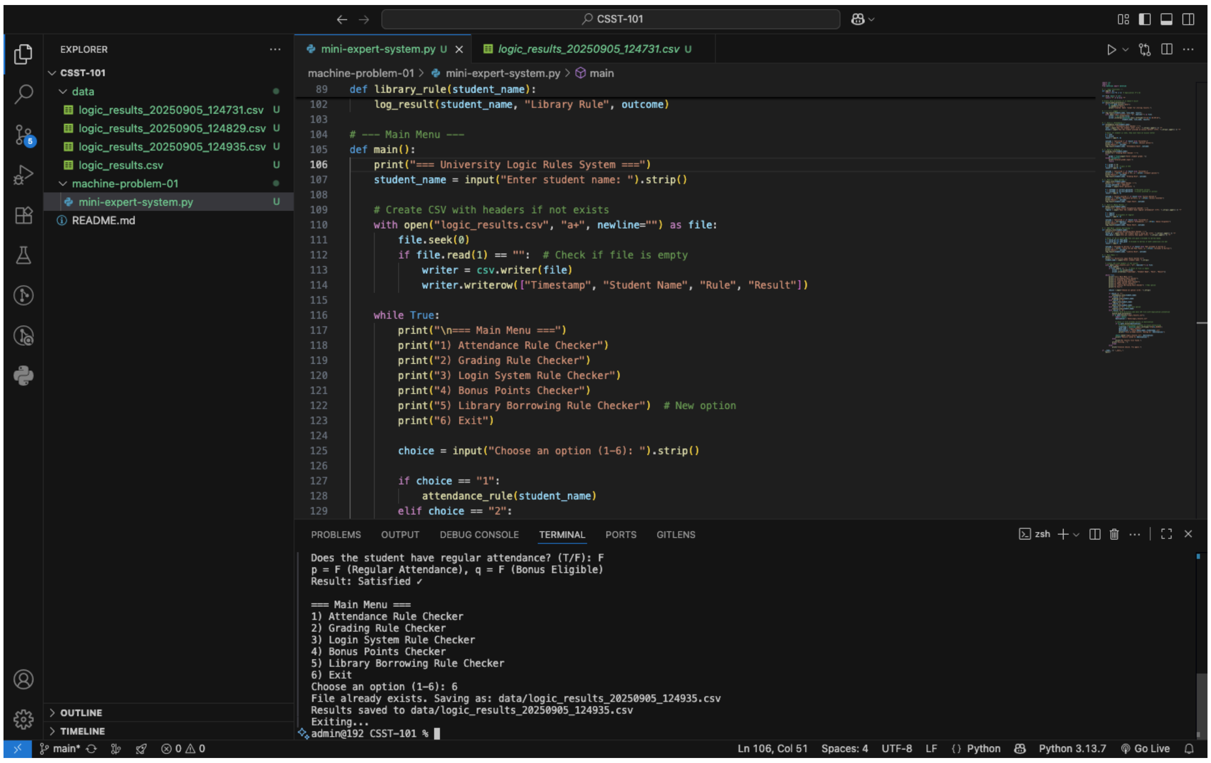
The system uses Python's logical operators to implement propositional logic:
- Implication Function: `impl(P, Q) = (not P) or Q`
- Truth Value Display: `tf(b)` converts boolean values to "T"/"F" format
- Modular Design: Each rule is implemented as a separate function for maintainability

## Conclusion

This mini expert system successfully demonstrates the practical application of logical reasoning in automated decision-making. The five implemented rules cover various university scenarios, from attendance policies to resource access control. The new Library Borrowing Rule specifically showcases how complex institutional policies can be encoded using compound logical conditions, making the system more representative of real-world expert systems used in educational institutions.

The logging and file management features ensure that all decisions are traceable and auditable, which is essential for administrative purposes and system transparency.

## Demonstration:



```python
def library_rule(student_name):
    log_result(student_name, "Library Rule", outcome)

# --- Main Menu ---
def main():
    print("=== University Logic Rules System ===")
    student_name = input("Enter student name: ").strip()

    # Create CSV with headers if not exists
    with open("logic_results.csv", "a+", newline="") as file:
        file.seek(0)
        if file.read(1) == "":  # Check if file is empty
            writer = csv.writer(file)
            writer.writerow(["Timestamp", "Student Name", "Rule", "Result"])

    while True:
        print("\n=== Main Menu ===")
        print("1) Attendance Rule Checker")
        print("2) Grading Rule Checker")
        print("3) Login System Rule Checker")
        print("4) Bonus Points Checker")
        print("5) Library Borrowing Rule Checker")  # New option
        print("6) Exit")

        choice = input("Choose an option (1-6): ").strip()

        if choice == "1":
            attendance_rule(student_name)
        elif choice == "2":
```

```
Does the student have regular attendance? (T/F): F
p = F (Regular Attendance), q = F (Bonus Eligible)
Result: Satisfied ✓

=== Main Menu ===
1) Attendance Rule Checker
2) Grading Rule Checker
3) Login System Rule Checker
4) Bonus Points Checker
5) Library Borrowing Rule Checker
6) Exit
Choose an option (1-6): 6
File already exists. Saving as: data/logic_results_20250905_124935.csv
Results saved to data/logic_results_20250905_124935.csv
Exiting...
admin@192 CSST-101 %
```

Name: Garin, Jeremy M.
Section: BSCS – 3A

CSST 101
Mr. Bernandino