**AI SAMADHAN PROBLEM FOUR**

**(AI Based Automated Exam Schedule Generation System)**

**“AI SAMADHAN**” is more than just coding competitions; this is **catalysts for innovation, learning, and collaboration**. It empower individuals and organizations to harness the power of AI to solve pressing challenges and drive technological progress. By participating in AI **SAMADHAN**, you contribute to the growth of the AI ecosystem and make a meaningful impact on society.

**Objective:**

To design and implement an automated system that optimizes the scheduling of **theory and**

**Practical exams** for both **regular and remedial** students in a university. The system will ensure

that:

1. Exams are scheduled within a maximum duration of **3 hours**.

2. The **minimum number of days** is used for the exam schedule.

3. There are **no conflicts** between students, classrooms, and exams.

4. The system generates **complex reports** to analyze the schedule's efficiency and resource

utilization.

**Scope:**

The system will cater to the needs of a university by automating the scheduling of exams. It will:

1. Schedule **theory and practical exams** for regular and remedial students.

2. Ensure that exams are scheduled within a maximum duration of 2 hours.

3. Minimize the number of days required for the exam schedule.

4. Avoid conflicts between students, classrooms, and exams.

5. Generate **complex reports** to provide insights into the schedule's efficiency and resource

utilization.

**Key Requirements:**

**1. Input Data:**

• **Courses and Exams:** List of courses, theory exams, and practical exams.

• **Students:** List of students enrolled in regular and remedial courses.

• **Classrooms:** List of available classrooms with their capacities.

• **Constraints:**

o Maximum exam duration: 2 hours.

o No overlapping exams for the same student.

o Classroom capacity must not be exceeded.

o Minimum number of days for the exam schedule.

o Maximum number of shifts: 3

• **Preferences:**

o Preferred time slots for exams (e.g., morning, afternoon).

**2. Output:**

• **Exam Schedule:** A conflict-free schedule for theory and practical exams.

• **Complex Reports:**

o Exam Schedule Report: Detailed schedule for each exam, student, and classroom.

o Resource Utilization Report: Analysis of classroom utilization and exam

distribution.

o Conflict and Constraint Violation Report: List of conflicts and violations (if any).

o Student Exam Schedule Report: Personalized schedule for each student.

**3. Constraints:**

• **Hard Constraints (Must be satisfied):**

o No overlapping exams for the same student.

o Classroom capacity must not be exceeded.

o Exams must be scheduled within 2 hours.

• **Soft Constraints (Preferred but not mandatory):**

o Minimize the number of days required for the exam schedule.

o Accommodate preferred time slots for exams.

**4. Functional Requirements:**

• Automatically generate an optimized exam schedule based on input data.

• Allow manual adjustments to the generated schedule.

• Handle last-minute changes (e.g., student absence, classroom unavailability).

• Generate and export complex reports in various formats (PDF, Excel).

**5. Non-Functional Requirements:**

• **Scalability:** The system should handle a large number of students, courses, and

classrooms.

• **Usability:** Intuitive interface for administrators to input data and view schedules and

reports.

• **Efficiency:** The system should generate the schedule and reports within a reasonable time

frame.

**Stakeholders:**

1. **Administrators:** Input data, generate schedules, manage changes, and view reports.

2. **Students:** Access their personalized exam schedules.

3. **Faculty:** View exam schedules and ensure no conflicts with their availability.

4. **Management:** Ensure efficient resource utilization and adherence to academic goals.

**Challenges:**

1. **Complexity:** Managing a large number of variables (students, courses, classrooms) and

constraints.

2. **Conflicts:** Resolving conflicts between students, classrooms, and exams.

3. **Optimization:** Minimizing the number of days required for the exam schedule.

4. **Reporting:** Generating accurate and insightful reports for decision-making.

**Deliverables:**

1. A software application or tool for optimized exam scheduling.

2. User manuals for administrators, students, and faculty.

3. Testing and validation reports to ensure the system meets all requirements.

**Success Criteria:**

1. The system generates a conflict-free exam schedule that meets all hard constraints.

2. The schedule uses the minimum number of days required.

3. Students and faculty report satisfaction with the schedule.

4. The system reduces the time and effort required for exam scheduling compared to manual methods.

5. The generated reports provide valuable insights into resource utilization and schedule

efficiency.

**Problem Definition Related Evaluation Criteria (Manual Checking):**

1. **Constraint Satisfaction:**

o Percentage of hard constraints satisfied (e.g., no overlapping exams, classroom

capacity respected).

o Percentage of soft constraints satisfied (e.g., preferred time slots, minimum

number of days).

2. **User Satisfaction:**

o Feedback from students and faculty on the exam schedule.

o Satisfaction scores based on surveys or interviews.

3. **Resource Utilization:**

o Classroom utilization rate (percentage of time classrooms are used).

4

o Exam distribution across days and time slots.

4. **Efficiency:**

o Time taken to generate the exam schedule.

o Time taken to generate reports.

5. **Report Accuracy:**

o Accuracy of exam schedule reports.

o Accuracy of resource utilization reports.

o Accuracy of conflict and constraint violation reports.

6. **Scalability:**

o Performance of the system with increasing numbers of students, courses, and

classrooms.

o Ability to handle multiple exam sessions (e.g., regular and remedial).

7. **Usability:**

o Ease of use for administrators, students, and faculty.

o Intuitiveness of the interface for inputting data and viewing schedules and reports.

**Programming Evaluation Criteria (Score generated by AI):**

1. Test Cases implemented in code. (higher the number higher the score)

2. Minimum required number of Files.

3. Repeat Lines Density (higher the number lower the score)

4. Code Complexity (higher the number lower the score)

5. Security Rating (higher the number lower the score)

6. Scale Rating (higher the number lower the score)

7. Reliability Rating (higher the number higher the score)

8. Bugs per File (higher the number lower the score)

9. Code Smells per File (higher the number lower the score)

10. Vulnerabilities per File (higher the number lower the score)

11. How much concepts of AI/ML used in your code. (higher the number higher the score)

12. How much Code written through AI (higher the number higher the score)

**Test Cases:**

**Test Case 1: Exam Schedule Generation**

• **Input:** Courses, students, classrooms, and constraints.

• **Expected Output:** A conflict-free exam schedule for theory and practical exams.

• **Evaluation:** Verify that all hard constraints are satisfied (e.g., no overlapping exams, classroom capacity respected).

**Test Case 2: Resource Utilization Report**

• **Input:** Generated exam schedule and classroom capacities.

• **Expected Output:** A report showing the utilization rate for each classroom.

• **Evaluation:** Verify that no classroom is overutilized or underutilized.

**Test Case 3: Conflict and Constraint Violation Report**

• **Input:** Generated exam schedule and constraints.

• **Expected Output:** A report listing any conflicts or constraint violations.

• **Evaluation:** Verify that the report accurately identifies and lists all conflicts and

violations.

**Test Case 4: Student Exam Schedule Report**

• **Input:** Generated exam schedule and student enrollments.

• **Expected Output:** A personalized exam schedule for each student.

• **Evaluation:** Verify that the schedule accurately reflects the exams for each student.

**Test Case 5: Manual Adjustments**

• **Input:** Manual adjustments to the generated schedule (e.g., changing an exam time).

• **Expected Output:** Updated schedule and reports reflecting the changes.

• **Evaluation:** Verify that the system handles manual adjustments correctly and updates the

reports accordingly.

**Test Case 6: Error Handling**

• **Input:** Invalid input data (e.g., missing student enrollments, incorrect classroom

capacities).

• **Expected Output:** Appropriate error messages and handling of invalid data.

• **Evaluation:** Verify that the system handles errors gracefully and provides meaningful

feedback.

**Example Reports:**

1. **Exam Schedule Report:**

o Exam: Introduction to AI (Theory)

o Date: 2023-10-15

o Time: 10:00 AM - 12:00 PM

o Classroom: Room 101

o Students: 50

2. **Resource Utilization Report:**

o Classroom: Room 101

o Utilization Rate: 90%

o Exams Scheduled: 5

3. **Conflict and Constraint Violation Report:**

o Conflict: Overlapping exams for Student John Doe.

o Constraint Violation: Classroom Room 101 over capacity for Exam Introduction

to AI (Theory).

4. **Student Exam Schedule Report:**

o Student: John Doe

o Exam: Introduction to AI (Theory)

o Date: 2023-10-15

o Time: 10:00 AM - 12:00 PM

Classroom: Room 101

ChatGpt

Copilot

Deepseek

Gemini

Grok

Minimax

Llama

Mistral

Mixtral

Fugaku

Gemma

Claude

Nemotron

Nova

Phi

Pixtral

Qwen