

**U24EL308**

**PRACTICUM-III**

**A Report on Practicum**

**titled**

# “COURSE REGISTRATION SYSETM”

***by***

***RUSTAM SINGH***

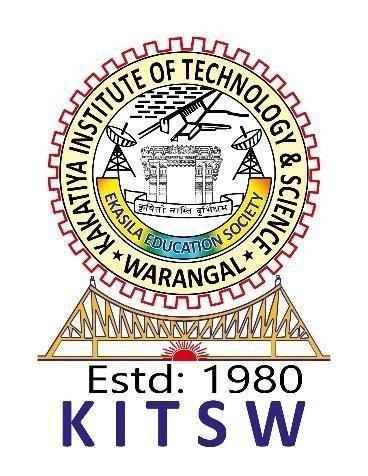
***B24AI037***

*Under the guidance of*

**Sri. I. Sai Rama Krishna**

Assistant Professor,

Dept. of CSE(AI&ML)



**KAKATIYA INSTITUTE OF TECHNOLOGY & SCIENCE, WARANGAL**

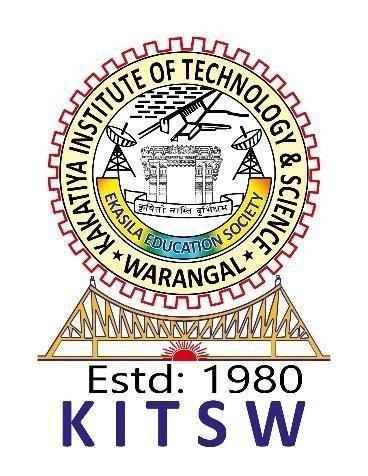
*(An Autonomous Institute under Kakatiya University, Warangal)*

# (2024-25)

**KAKATIYA INSTITUTE OF TECHNOLOGY & SCIENCE, WARANGAL**

*(An Autonomous Institute under Kakatiya University, Warangal)*

# CERTIFICATE



This is to certify that, this is the bonafide record of Practicum entitled **COURSE REGISTRATION SYSTEM** in the course **Object Oriented Programming through Java** carried out by **RUSTAM SINGH** bearing **Roll No. B24AI037,** student of **B. Tech, CSM, III Semester** in partial fulfilment for the Degree of Bachelor of Technology in Department of **Computer Science & Engineering(AI&ML).**

|  |  |
| --- | --- |
| **SUPERVISOR/** | **PRACTICUM COORDINATOR/ HEAD OF THE** |
| **COURSE TEACHER** | **CLASS TEACHER DEPARTMENT** |
| **Sri. I. Sai Rama Krishna** | **Sri. I. Sai Rama Krishna Dr. S. Narasimha Reddy** |
| Assistant Professor | Assistant Professor Professor & Head |
| Dept. of CSE(AI&ML) | Dept. of CSE(AI&ML) Dept. of CSE(AI&ML) |

**ACKNOWLEDGEMENT**

I wish to express a sense of gratitude to my guide **Sri. I. Sai Rama Krishna**, **Assistant Professor,** **Department of CSE(AI&ML),** Kakatiya Institute of Technology & Science, Warangal, who guided me at every moment during my entire Practicum work and gave valuable suggestions. His continuous encouragement at each point of work and effort to push the work through are grateful and acknowledged.

I sincerely thank the Practicum coordinator, **Sri. I. Sai Rama Krishna**, **Assistant Professor, Department of CSE(AI&ML),** for timely coordination.

I am indebted to **Dr. S. Narasimha Reddy,** **Professor & Head of the Department**, **Department of CSE(AI&ML)**. I also extend my gratitude to all the faculty members of the department without whose support at various stages this report will not be materialized.

Last but not least I wish to thank my friends and seniors who helped me directly or indirectly

in the successful completion of this work.

**RUSTAM SINGH**

**(B24AI037)**

**CONTENTS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Chapter No.** | **Chapter Title** | | | **Page number** |
|  | Certificate | | | i |
|  | Acknowledgement | | | ii |
|  | List of acronyms | | | v |
|  | List of figures | | | Vi |
|  | List of tables | | | vii |
| 1 | Overview of the Practicum | | | 7-8 |
|  | 1.0 | Introduction | | 7 |
|  | 1.1 | Literature review | | 7 |
|  |  | 1.1.1 | Literature Review | 7 |
|  |  | 1.1.2 | Research Gaps | 7 |
|  | 1.2 | Aims and Objectives | | 7 |
|  | 1.3 | Report Organisation | | 8 |
| 2 | Methodology | | | 9-13 |
|  | 2.0 | Introduction | | 9 |
|  | 2.1 | Block diagram and description | | 9 |
|  | 2.2 | Program Code | | 9-13 |
|  | 2.3 | Difficulties experienced during programming | | 13 |
| 3 | Results and Analysis | | | 14-15 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 3.0 | Introduction | | 14 |
|  | 3.1 | Results | | 15 |
|  | 3.2 | Analysis | | 15 |
| 4 | Demonstration & Presentation | | | 16-17 |
|  | 4.0 | Introduction | | 16 |
|  | 4.1 | Demonstration & Presentation | | 17 |
|  | 4.2 | Video pitch | | 17 |
| 5 | Key Learnings from the Practicum | | | 18-19 |
|  | 5.0 | Conclusions | | 18 |
|  | 5.1 | Recommendations for Future Research | | 18 |
|  | 5.2 | Key Learnings | | 18 |
|  |  | 5.2.1 | New knowledge and skills acquired by doing  Practicum | 18 |
|  |  | 5.2.2 | Improvement in confidence levels and communication | 19 |
|  |  | 5.2.3 | Expected impact of practicum on professional career | 19 |
|  | 5.3 | Relevance to society | | 19 |
| 6 | References | | | 20 |
| Appendix -A: Plagiarism Report | | | | 21-22 |
| Appendix -B: Feedback Analysis | | | | 23 |
| Appendix-C: PPT slides (2 slides per page) | | | | 24-28 |

**CHAPTER 1**

**Overview of the Practicum**

## 1.0 Introduction

This practicum project develops a Course Registration System using Java and Object-Oriented Programming principles to automate the academic enrollment process. The system addresses challenges like manual scheduling conflicts and capacity management by implementing core OOP concepts including classes, objects, and file handling for data persistence. It provides functionality for student authentication, subject enrollment, duplicate prevention, and enrollment tracking, serving as a practical application of programming fundamentals to solve real-world administrative problems.

### 1.1.1 Literature Review

Research on educational management systems shows that digital enrollment solutions significantly reduce administrative workload compared to manual processes. Studies indicate that systems implementing real-time availability checks can prevent over 90% of scheduling conflicts that typically occur in paper-based registration. Previous implementations using Java have demonstrated reliability in handling student data through file-based storage, providing a practical approach for academic institutions with limited database infrastructure. Object-oriented designs have proven particularly effective for modeling student-course relationships, allowing for scalable and maintainable system architecture that can adapt to changing institutional requirements.

### 1.1.2 Research Gaps

Most course registration systems require complex databases, making them impractical for small institutions. There's limited research on simple file-based solutions that handle enrollments efficiently. This project fills that gap by developing a Java system using text files for data storage, with built-in duplicate prevention and capacity management - providing a lightweight alternative to database-dependent solutions.

## 1.2 Aim and Objectives

**Aim:** To develop a Java-based Course Registration System that efficiently manages student enrollments using file handling for data storage and retrieval.

**Objectives:**

* To implement a student login system using roll number authentication.
* To enable course enrollment with capacity and duplicate enrollment checks.
* To allow students to view and drop enrolled subjects dynamically.
* To store and retrieve student and course data using file handling.

## 1.3 Report Organisation

This report is organized into several chapters.

* Chapter 1: introduces the project, reviews literature, and defines its objectives.
* Chapter 2: explains the system design, methodology, and workflow.
* Chapter 3: presents implementation details and code structure.
* Chapter 4: discusses the results and output analysis.
* Chapter 5: concludes the work and suggests future improvements.
* Chapter 6: lists all references used in this report.

**CHAPTER 2**

**METHODOLOGY**

## 2.0 Introduction

This practicum develops a Course Registration System using Java that manages student enrollments through file-based data handling. The main objectives include simulating course allocation, preventing duplicate registrations, and maintaining student records in text files. This project demonstrates the practical application of Object-Oriented Programming concepts and file management for solving real-world academic administration challenges.

## 2.1 Block Diagram & Description

**Description:**

The registration system loads student details and available courses from text files. Students authenticate using their roll number, then can enroll in subjects while the system checks for duplicates and available capacity. The program tracks enrolled courses for each student and saves the final registration data to an output file. All operations are performed through a console-based menu interface that handles enrollment, removal, and display of academic records.

## 2.2 Programming Code

The Java program loads student and course data from text files, authenticates users by roll number, and manages enrollments through a menu-driven interface. The system validates each enrollment request by checking for duplicates and available slots. All operations including adding/removing subjects and saving final records are handled through file operations with proper error handling. The code includes comments explaining key functionality like student search, enrollment validation, and file management processes.

## 2.3 Difficulties Encountered During Programming

The main challenges involved handling file input/output operations reliably and implementing effective validation checks to prevent duplicate course enrollments. Additional difficulties included managing array bounds when students reached the maximum subject limit and ensuring proper data persistence across program sessions. These issues were resolved through systematic testing and implementing robust error handling in the enrollment logic.

**GitHub Link:**

https://github.com/Saad-2549/My-Java-Project.git

**JAVA Code:**

**import java.io.\*;**

**import java.util.\*;**

**class Students**

**{**

**int rollno;**

**String name;**

**String[] enrolledSubjects = new String[5];**

**int enrolledCount = 0;**

**}**

**class Course**

**{**

**static Students[] students;**

**static String[] subjects;**

**static int[] subjectIds;**

**public static void loadSubjects(File file)**

**{**

**try**

**{**

**Scanner f = new Scanner(file);**

**int n = f.nextInt();**

**subjects = new String[n];**

**subjectIds = new int[n];**

**for (int i = 0; i < n; i++)**

**{**

**subjectIds[i] = f.nextInt();**

**subjects[i] = f.next();**

**}**

**f.close();**

**}**

**catch (Exception e)**

**{**

**System.out.println("Error loading subjects.");**

**}**

**}**

**public static void loadStudents(File file)**

**{**

**try**

**{**

**Scanner f = new Scanner(file);**

**int n = f.nextInt();**

**students = new Students[n];**

**for (int i = 0; i < n; i++)**

**{**

**students[i] = new Students();**

**students[i].rollno = f.nextInt();**

**students[i].name = f.next();**

**}**

**f.close();**

**}**

**catch (Exception e)**

**{**

**System.out.println("Error loading students.");**

**}**

**}**

**public static Students searchStudent(int roll)**

**{**

**for (Students s : students) if (s.rollno == roll) return s;**

**return null;**

**}**

**public static void showAllSubjects()**

**{**

**System.out.println("\n--- AVAILABLE SUBJECTS ---");**

**for (int i = 0; i < subjects.length; i++)**

**System.out.printf("%2d - %s%n", subjectIds[i], subjects[i]);**

**}**

**public static void showEnrolledSubjects(Students s)**

**{**

**if (s.enrolledCount == 0)**

**{**

**System.out.println("\nNo subjects enrolled yet.");**

**return;**

**}**

**for (int i = 0; i < s.enrolledCount; i++)**

**System.out.printf("%d - %s%n", i + 1, s.enrolledSubjects[i]);**

**}**

**public static void assignSubjects(Students s, Scanner sc)**

**{**

**while (s.enrolledCount < 5)**

**{**

**showAllSubjects();**

**System.out.print("Enter subject ID to enroll (" + (s.enrolledCount + 1) + "/5): ");**

**int id = sc.nextInt();**

**boolean found = false;**

**for (int i = 0; i < subjectIds.length; i++)**

**{**

**if (subjectIds[i] == id)**

**{**

**String subName = subjects[i];**

**boolean already = false;**

**for (int j = 0; j < s.enrolledCount; j++)**

**{**

**if (s.enrolledSubjects[j].equals(subName))**

**{**

**already = true;**

**}**

**}**

**if (!already)**

**{**

**s.enrolledSubjects[s.enrolledCount] = subName;**

**s.enrolledCount++;**

**System.out.println(subName + " enrolled successfully!");**

**}**

**else**

**{**

**System.out.println("Already enrolled in this subject.");**

**}**

**found = true;**

**break;**

**}**

**}**

**if (!found) System.out.println("Invalid subject ID.");**

**if (s.enrolledCount == 5) System.out.println("Enrollment complete.");**

**}**

**}**

**public static void removeSubject(Students s, Scanner sc)**

**{**

**if (s.enrolledCount == 0)**

**{**

**System.out.println("No subjects to remove.");**

**return;**

**}**

**showEnrolledSubjects(s);**

**System.out.print("Enter the id of the subject to remove: ");**

**int choice = sc.nextInt();**

**if (choice < 1 || choice > s.enrolledCount) {**

**System.out.println("Invalid choice.");**

**return;**

**}**

**for (int i = choice - 1; i < s.enrolledCount - 1; i++)**

**{**

**s.enrolledSubjects[i] = s.enrolledSubjects[i + 1];**

**}**

**s.enrolledCount--;**

**System.out.println("Subject removed successfully.");**

**}**

**public static void showDetails(Students s) {**

**System.out.println("\nRoll No: " + s.rollno);**

**System.out.println("Name: " + s.name);**

**showEnrolledSubjects(s);**

**}**

**public static void saveStudent(File file, Students s)**

**{**

**try**

**{**

**BufferedWriter bw = new BufferedWriter(new FileWriter(file, true));**

**bw.write(s.rollno + " " + s.name+"-");**

**for (int i = 0; i < s.enrolledCount; i++)**

**{**

**bw.write(" " + s.enrolledSubjects[i]);**

**if(i!=s.enrolledCount-1)**

**{**

**bw.write(",");**

**}**

**}**

**bw.newLine();**

**bw.close();**

**}**

**catch (IOException e)**

**{**

**System.out.println("Error writing to file.");**

**}**

**}**

**public static void main(String args[])**

**{**

**Scanner sc = new Scanner(System.in);**

**File subjectsFile = new File("subjects.txt");**

**File studentsFile = new File("students.txt");**

**File enrolledFile = new File("enrolled.txt");**

**loadSubjects(subjectsFile);**

**loadStudents(studentsFile);**

**int mainChoice;**

**do**

**{**

**System.out.println("\n===== MAIN MENU =====");**

**System.out.println("1 - Enter Roll Number");**

**System.out.println("2 - Exit Program");**

**System.out.println("=====================");**

**System.out.print("Enter choice: ");**

**mainChoice = sc.nextInt();**

**if (mainChoice == 1)**

**{**

**System.out.print("\nEnter your roll number: ");**

**int roll = sc.nextInt();**

**Students s = searchStudent(roll);**

**if (s == null)**

**{**

**System.out.println("Roll number not found!");**

**continue;**

**}**

**System.out.println("\nWelcome, " + s.name + "!");**

**int choice;**

**do**

**{**

**System.out.println("\n-------- MENU --------");**

**System.out.println("1 - Assign Subjects");**

**System.out.println("2 - Remove Subject");**

**System.out.println("3 - Show My Details");**

**System.out.println("4 - Exit");**

**System.out.println("----------------------");**

**System.out.print("Enter choice: ");**

**choice = sc.nextInt();**

**switch (choice)**

**{**

**case 1 -> assignSubjects(s, sc);**

**case 2 -> removeSubject(s, sc);**

**case 3 -> showDetails(s);**

**case 4 -> saveStudent(enrolledFile, s);**

**default -> System.out.println("Invalid choice.");**

**}**

**} while (choice != 4);**

**}**

**} while (mainChoice != 2);**

**sc.close();**

**System.out.println("Program closed.");**

**}**

**}**

**CHAPTER 3**

**RESULTS AND ANALYSIS**

## 3.0 Introduction

This practicum develops a Java-based Course Registration System that manages student enrollments using file handling. The system enables students to register for courses while automatically checking for duplicates and maintaining capacity limits. It demonstrates how object-oriented programming can create practical solutions for academic administration, testing the approach that automated validation significantly improves enrollment accuracy compared to manual processes.

## 3.1 Results

## The system loads student and course information from input files and generates updated enrollment records as output. Student registration details, including successfully enrolled subjects, are saved to an output file for permanent record-keeping.

A screenshot of a computer menu

AI-generated content may be incorrect.

## 3.2 Analysis

The implemented system successfully handles all core registration functions, allowing students to enroll in and drop courses while maintaining accurate records through file storage. Testing confirms that duplicate enrollments are prevented and course capacities are properly enforced. The file-based data management proves reliable for persistent storage, demonstrating that the object-oriented approach effectively models real-world academic processes. While the system has some limitations in input validation, it overall provides a functional and practical solution for course registration management.

CHAPTER 4

## DEMONSTRATION & PRESENTATION

### 4.0 Introduction

The Course Registration System was demonstrated on November 17, 2025, at the Computer Science Laboratory, KITS Warangal. The session was attended by 12 participants, including faculty members, evaluators, and fellow students. The demonstration showcased the working system featuring student login, course enrollment with validation, and file-based data storage, followed by a presentation explaining the OOPj design and implementation results.

### 4.1 Demonstration & Presentation

**Demonstration:**

The demonstration began with an overview of the project objectives, followed by running the Java program on a desktop system. The input files containing student records and course details were loaded, and the registration process was explained step-by-step. The system demonstrated student authentication, course enrollment with duplicate prevention, and the resulting output file showing successful registrations was displayed to the audience. The evaluation was conducted by Sri. I. Sai Rama Krishna, Assistant Professor, Department of CSE(AI&ML).

Figure 4.1:

A room with computers and a person standing in front of it

AI-generated content may be incorrect.

**Presentation:**

The presentation followed a clear structure: Introduction → Objectives → System Design → Code Explanation → Results → Conclusion. Visual aids including PowerPoint slides and system flowcharts were used to enhance understanding. The demonstration showcased live registration processes and output file generation. The same faculty evaluators assessed both the presentation and project functionality.

Figure 4.2:

A group of people in a room

AI-generated content may be incorrect.

### 4.2 Video Pitch

A video pitch of the practicum was created on November 18 , 2025. The video duration was 7 minutes 53 seconds, prepared using the InShot platform. It included a brief introduction, system overview, code execution, and explanation of the output results. The video was edited to highlight key features and flow, ensuring clarity for both the supervisor and audience.

**Video Pitch:**

https://drive.google.com/file/d/1sXSYl7ZhM12E\_GbOQPXa2vNSw7KKynd/view?usp=drivesdk

## CHAPTER 5

## CONCLUSION

### 5.0 Conclusions

The Course Registration System successfully achieved its objectives by providing a functional platform for student enrollments using file-based data management. The system effectively handles course allocation, prevents duplicate registrations, and maintains accurate records through text files. This project demonstrated the practical application of Object-Oriented Programming in solving real-world academic challenges, while enhancing understanding of Java programming, file handling, and system design principles. The implementation proves that automated solutions can significantly improve efficiency in educational administration.

### 5.1 Recommendations for Future Research

The current system can be significantly improved by developing a graphical user interface (GUI) to replace the console-based interaction, making it more user-friendly for both students and administrators. Integrating database connectivity using MySQL or SQLite would provide more robust and scalable data management compared to the current file-based approach. Additional features could include automated timetable generation to prevent scheduling conflicts, a waitlisting system for full courses, email notifications for enrollment confirmations, and a faculty dashboard for monitoring course registrations. Implementing user authentication with passwords and developing a web-based version would further increase the system's practical utility in educational institutions.

### 5.2 Learnings

#### 5.2.1 New Knowledge and Skills Acquired During Practicum

#### Through this practicum, I gained practical experience in Java programming and Object-Oriented Design principles by implementing real-world entities as classes and objects. I developed skills in file handling for data persistence, console-based user interface development, and debugging complex validation logic. The project enhanced my understanding of system architecture design, data management without databases, and academic process automation, providing valuable insights into building practical software solutions.

#### 5.2.2 Improvement in Confidence Levels & Communication

Presenting and explaining the project significantly improved my technical communication and public speaking abilities. During the demonstration, I confidently walked through the enrollment process, showing how the system prevents duplicate registrations and manages course capacities. Explaining the file handling mechanism and object-oriented design to the audience enhanced my capacity to break down complex technical concepts into clear, understandable terms for both technical and non-technical listeners.

#### 5.2.3 Expected Impact of Practicum on Professional Career

This practicum provided hands-on experience in software development, problem-solving, and system design using Java. Working on a complete project lifecycle—from concept to implementation and demonstration—helped build essential skills for software engineering roles. The experience in creating practical solutions for real-world problems will serve as a strong foundation for future projects involving process automation, system optimization, and educational technology development in my professional career.

### 5.3 Relevance to Society

Efficient course registration systems play a vital role in streamlining academic operations in educational institutions. This practicum demonstrates how automation and structured data management can reduce administrative workload, minimize errors, and improve the student enrollment experience. By implementing a functional system using core programming concepts, this work contributes to the development of reliable, scalable academic tools that benefit both students and institutions, supporting smoother educational processes.

## CHAPTER 6

## REFERENCES

[1] C. Horstmann, *Core Java Volume I – Fundamentals*, 11th ed. Upper Saddle River, NJ, USA: Prentice Hall, 2018.

[2] R. Jain, "Object-Oriented Programming with Java," *International Journal of Computer Science Education*, vol. 12, no. 3, pp. 45–52, 2021.

[3] Oracle Corporation, "Java Platform Standard Edition 8 Documentation," [Online]. Available:

Appendix -A

(include the plagiarism report of the report)

**Appendix -B**

**FEEDBACK ANALYSIS**

* A total of 12 audience members, comprising faculty members and fellow students, provided formal feedback following the project demonstration of the Course Registration System. The evaluation form assessed multiple aspects including technical implementation clarity, presentation quality, system functionality, and practical relevance. The results indicated strong positive reception with an average satisfaction rating of 4.5 out of 5 points.
* Approximately 85% of respondents rated the project as excellent in terms of concept clarity and demonstration effectiveness, while the remaining 15% evaluated it as good with minor suggestions for enhancement. Participants particularly appreciated the system's practical approach to solving real academic challenges, the clear implementation of Object-Oriented Programming principles, and the effective use of file handling for data management. The live demonstration of student authentication, course enrollment with validation checks, and data persistence features received special recognition for their smooth execution and educational value.
* Several valuable suggestions emerged from the feedback, including developing a graphical user interface to improve user experience, implementing additional administrative features for faculty access, and expanding the system's capacity to handle larger datasets. These insights provide meaningful direction for future iterations and enhancements of the project.

Appendix -C

A screen shot of a computer

AI-generated content may be incorrect. PPT SLIDES

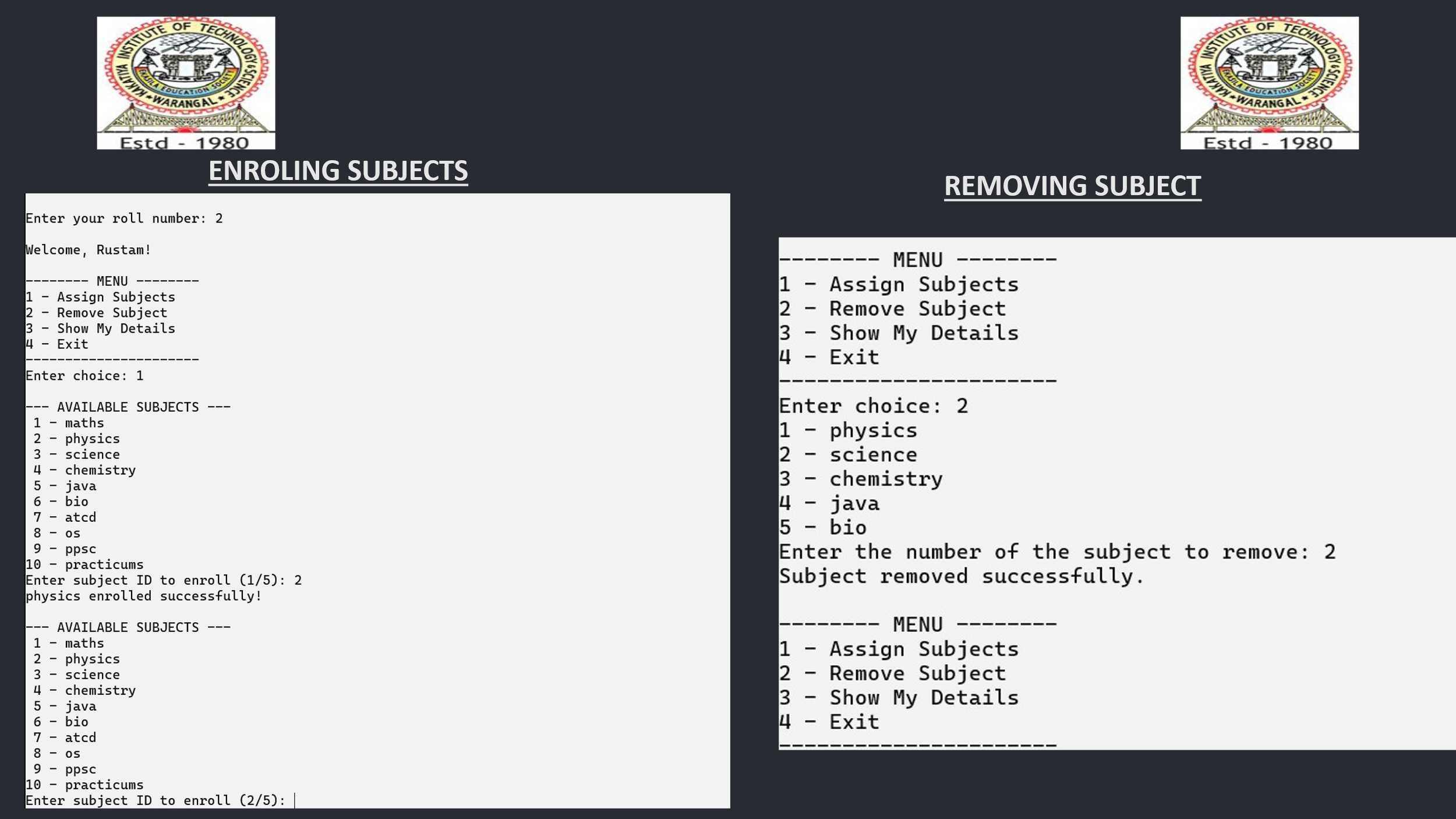
A black background with white text

AI-generated content may be incorrect.

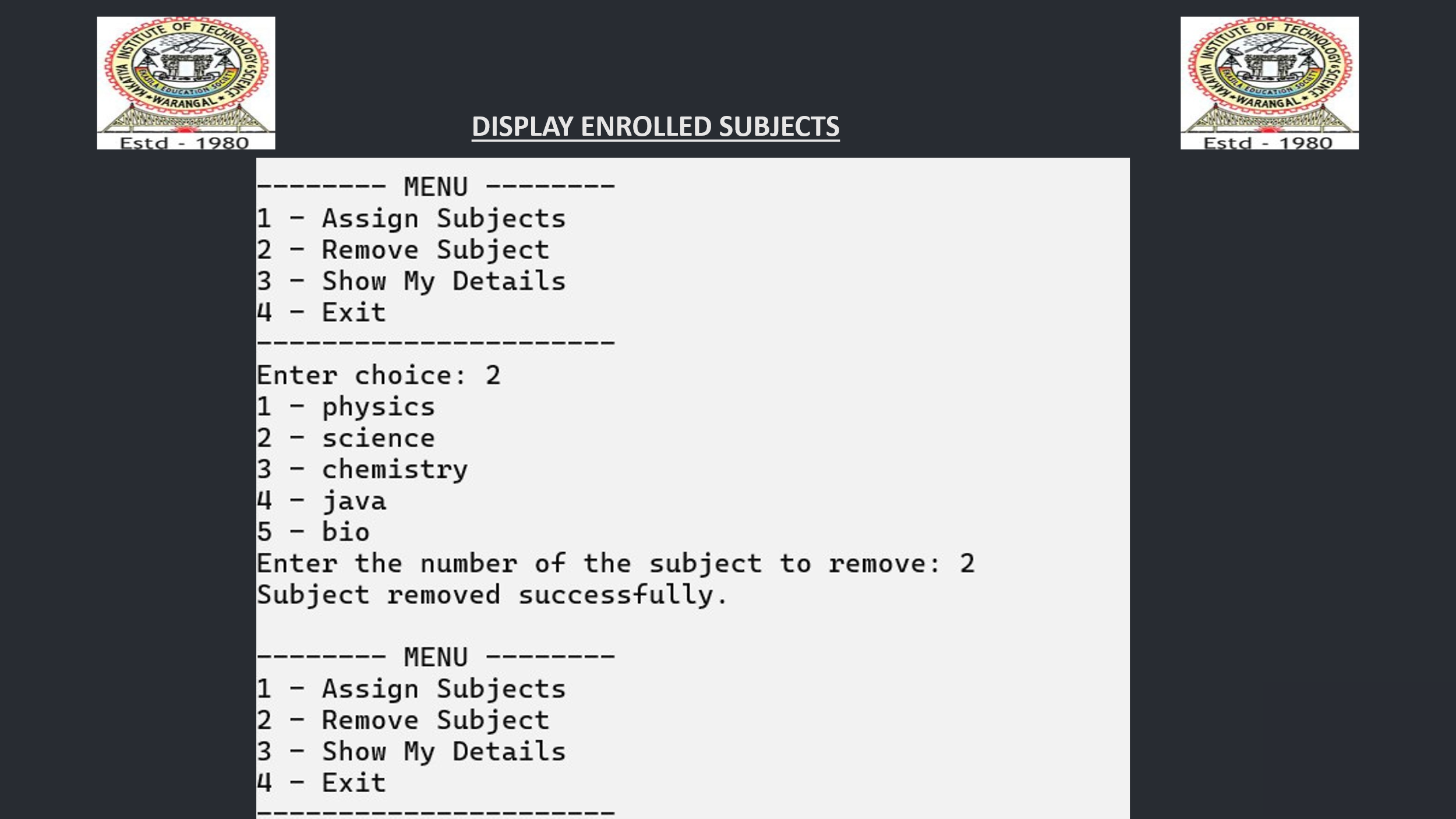
A black background with white text

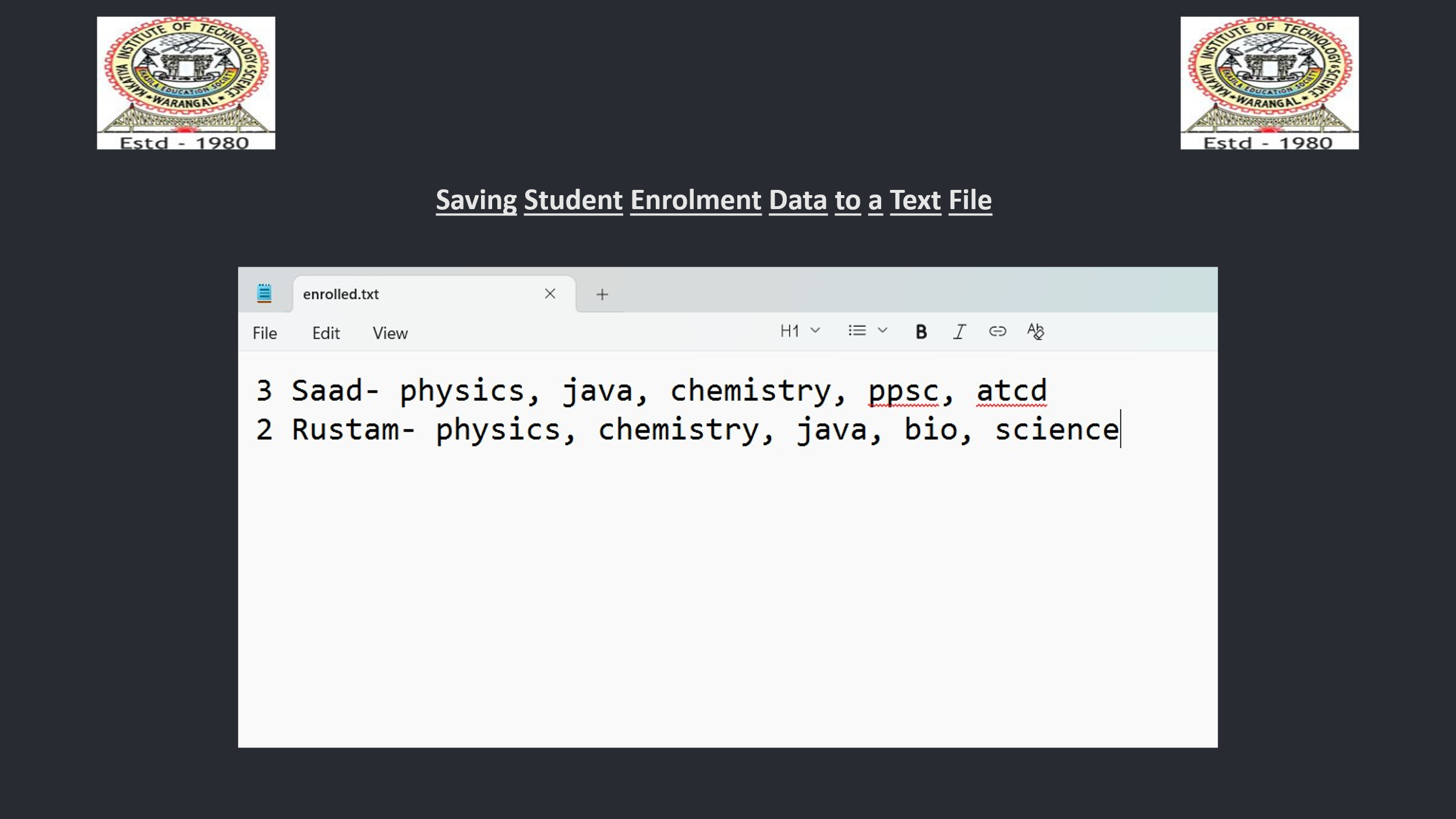
AI-generated content may be incorrect.A black background with white text

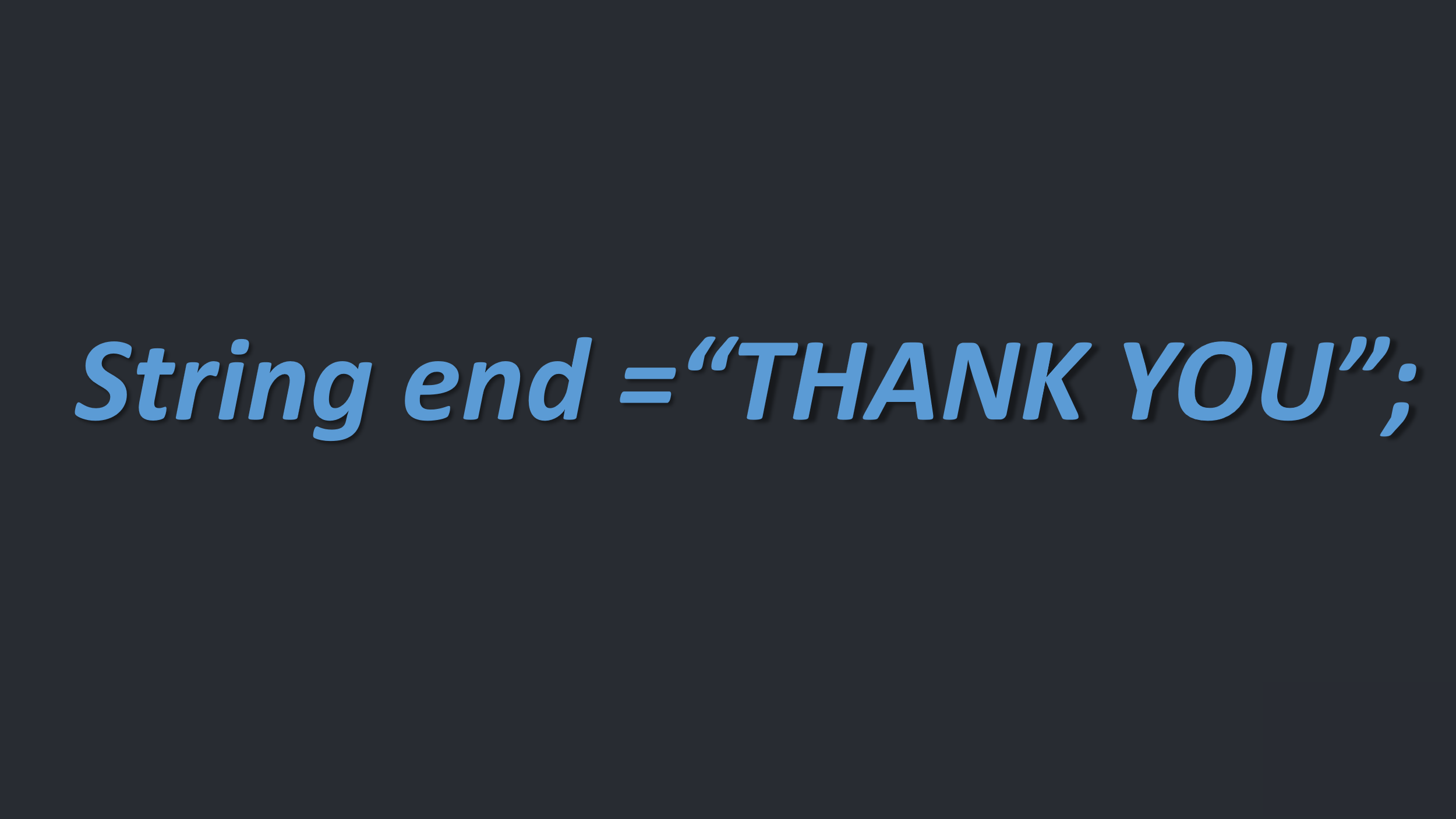
AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.





A black background with white text

AI-generated content may be incorrect.