#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#define MAX\_COURSES 100

#define MAX\_STUDENTS 500

#define MAX\_FACULTY 100

#define MAX\_GRADES 100

typedef struct

{

char name[50];

char course\_id[20];

int credits;

} Course;

typedef struct

{

char name[50];

char student\_id[20];

Course courses\_enrolled[MAX\_COURSES];

int num\_courses;

float grades[MAX\_COURSES]; // Store grades for courses

} Student;

typedef struct

{

char name[50];

char faculty\_id[20];

Course courses\_taught[MAX\_COURSES];

int num\_courses;

} Faculty;

typedef struct

{

char name[50];

char admin\_id[20];

} Admin;

// Function Prototypes

void display\_courses(Course courses[], int num\_courses);

void enroll\_course(Student \*student, Course course);

void drop\_course(Student \*student, char course\_id[]);

void view\_schedule(Student student);

void manage\_courses(Faculty \*faculty);

void handle\_grades(Faculty faculty);

void handle\_attendance(Faculty faculty);

void admin\_manage\_students();

void admin\_manage\_faculty();

void admin\_manage\_courses();

void view\_faculties\_and\_students(Student students[], int num\_students, Faculty faculty[], int num\_faculty);

// Global Data

Course courses[MAX\_COURSES] = {

{"Math 101", "Math101", 3},

{"Physics 202", "Physics202", 4},

{"Chemistry 303", "Chemistry303", 3}};

int num\_courses = 3;

Student students[MAX\_STUDENTS];

int num\_students = 0;

Faculty faculty[MAX\_FACULTY];

int num\_faculty = 0;

Admin admins[MAX\_FACULTY];

int num\_admins = 0;

int main()

{

// Initialize test data

strcpy(students[0].name, "Alice");

strcpy(students[0].student\_id, "S001");

students[0].num\_courses = 0;

num\_students++;

strcpy(faculty[0].name, "Dr. Smith");

strcpy(faculty[0].faculty\_id, "F001");

faculty[0].num\_courses = 0;

num\_faculty++;

strcpy(admins[0].name, "AdminUser ");

strcpy(admins[0].admin\_id, "A001");

num\_admins++;

int choice;

while (1)

{

printf("\n--- Course Registration System ---\n");

printf("1. Student Login\n");

printf("2. Faculty Login\n");

printf("3. Admin Login\n");

printf("4. Exit\n");

printf("Enter Your Choice: ");

scanf("%d", &choice);

switch (choice)

{

case 1:

{

char student\_id[20];

printf("Enter Student ID: ");

scanf("%s", student\_id);

// Find student

int student\_index = -1;

for (int i = 0; i < num\_students; i++)

{

if (strcmp(students[i].student\_id, student\_id) == 0)

{

student\_index = i;

break;

}

}

if (student\_index != -1)

{

// Student Menu

while (1)

{

printf("\n--- Student Menu ---\n");

printf("1. View Schedule\n");

printf("2. Enroll in Course\n");

printf("3. Drop Course\n");

printf("4. Logout\n");

printf("Enter Your Choice: ");

scanf("%d", &choice);

switch (choice)

{

case 1:

view\_schedule(students[student\_index]);

break;

case 2:

{

int course\_index;

display\_courses(courses, num\_courses);

printf("Enter Course Index to Enroll: ");

scanf("%d", &course\_index);

course\_index--; // Adjust for 0-based index

if (course\_index >= 0 && course\_index < num\_courses)

{

enroll\_course(&students[student\_index], courses[course\_index]);

}

else

{

printf("Invalid Course Index!\n");

}

break;

}

case 3:

{

char course\_id[20];

printf("Enter Course ID to Drop: ");

scanf("%s", course\_id);

drop\_course(&students[student\_index], course\_id);

break;

}

case 4:

goto exit\_student;

default:

printf("Invalid Choice!\n");

}

}

}

else

{

printf("Invalid Student ID!\n");

}

exit\_student:;

break;

}

case 2:

{

char faculty\_id[20];

printf("Enter Faculty ID: ");

scanf("%s", faculty\_id);

// Find faculty

int faculty\_index = -1;

for (int i = 0; i < num\_faculty; i++)

{

if (strcmp(faculty[i].faculty\_id, faculty\_id) == 0)

{

faculty\_index = i;

break;

}

}

if (faculty\_index != -1)

{

// Faculty Menu

while (1)

{

printf("\n--- Faculty Menu ---\n");

printf("1. Manage Courses\n");

printf("2. Handle Grades\n");

printf("3. Handle Attendance\n");

printf("4. Logout\n");

printf("Enter Your Choice: ");

scanf("%d", &choice);

switch (choice)

{

case 1:

manage\_courses(&faculty[faculty\_index]);

break;

case 2:

handle\_grades(faculty[faculty\_index]);

break;

case 3:

handle\_attendance(faculty[faculty\_index]);

break;

case 4:

goto exit\_faculty;

default:

printf("Invalid Choice!\n");

}

}

}

else

{

printf("Invalid Faculty ID!\n");

}

exit\_faculty:;

break;

}

case 3:

{

char admin\_id[20];

printf("Enter Admin ID: ");

scanf("%s", admin\_id);

// Find admin

int admin\_index = -1;

for (int i = 0; i < num\_admins; i++)

{

if (strcmp(admins[i].admin\_id, admin\_id) == 0)

{

admin\_index = i;

break;

}

}

if (admin\_index != -1)

{

while (1)

{

printf("\n--- Admin Menu ---\n");

printf("1. Manage Students\n");

printf("2. Manage Faculty\n");

printf("3. Manage Courses\n");

printf("4. View all faculties and students\n");

printf("5. Logout\n");

printf("Enter Your Choice: ");

scanf("%d", &choice);

switch (choice)

{

case 1:

admin\_manage\_students();

break;

case 2:

admin\_manage\_faculty();

break;

case 3:

admin\_manage\_courses();

break;

case 4:

view\_faculties\_and\_students(students, num\_students, faculty, num\_faculty);

case 5:

goto exit\_admin;

default:

printf("Invalid Choice!\n");

}

}

}

else

{

printf("Invalid Admin ID!\n");

}

exit\_admin:;

break;

}

case 4:

exit(0);

default:

printf("Invalid Choice! Please try again.\n");

}

}

}

// Function Definitions

void display\_courses(Course courses[], int num\_courses)

{

printf("\n--- Available Courses ---\n");

for (int i = 0; i < num\_courses; i++)

{

printf("%d. %s (%s) - %d credits\n", i + 1, courses[i].name, courses[i].course\_id, courses[i].credits);

}

}

void enroll\_course(Student \*student, Course course)

{

if (student->num\_courses < MAX\_COURSES)

{

student->courses\_enrolled[student->num\_courses++] = course;

printf("Enrolled in %s successfully!\n", course.name);

}

else

{

printf("Cannot enroll, maximum courses reached!\n");

}

}

void drop\_course(Student \*student, char course\_id[])

{

for (int i = 0; i < student->num\_courses; i++)

{

if (strcmp(student->courses\_enrolled[i].course\_id, course\_id) == 0)

{

for (int j = i; j < student->num\_courses - 1; j++)

{

student->courses\_enrolled[j] = student->courses\_enrolled[j + 1];

}

student->num\_courses--;

printf("Dropped course %s successfully!\n", course\_id);

return;

}

}

printf("Course not found in enrolled list!\n");

}

void view\_schedule(Student student)

{

printf("\n--- Your Schedule ---\n");

for (int i = 0; i < student.num\_courses; i++)

{

printf("%s (%s) - %d credits\n", student.courses\_enrolled[i].name, student.courses\_enrolled[i].course\_id, student.courses\_enrolled[i].credits);

}

}

void manage\_courses(Faculty \*faculty)

{

printf("\n--- Manage Courses ---\n");

int choice;

printf("1. Add a Course\n2. Remove a Course\nEnter Your Choice: ");

scanf("%d", &choice);

if (choice == 1)

{

if (num\_courses < MAX\_COURSES)

{

char name[50], course\_id[20];

int credits;

printf("Enter Course Name: ");

scanf("%s", name);

printf("Enter Course ID: ");

scanf("%s", course\_id);

printf("Enter Credits: ");

scanf("%d", &credits);

strcpy(courses[num\_courses].name, name);

strcpy(courses[num\_courses].course\_id, course\_id);

courses[num\_courses].credits = credits;

num\_courses++;

printf("Course added successfully!\n");

}

else

{

printf("Cannot add more courses, limit reached!\n");

}

}

else if (choice == 2)

{

char course\_id[20];

printf("Enter Course ID to Remove: ");

scanf("%s", course\_id);

int course\_index = -1;

for (int i = 0; i < num\_courses; i++)

{

if (strcmp(courses[i].course\_id, course\_id) == 0)

{

course\_index = i;

break;

}

}

if (course\_index != -1)

{

for (int i = course\_index; i < num\_courses - 1; i++)

{

courses[i] = courses[i + 1];

}

num\_courses--;

printf("Course removed successfully!\n");

}

else

{

printf("Course not found!\n");

}

}

else

{

printf("Invalid Choice!\n");

}

}

void handle\_grades(Faculty faculty)

{

char student\_id[20], course\_id[20];

float grade;

printf("Enter Student ID: ");

scanf("%s", student\_id);

printf("Enter Course ID: ");

scanf("%s", course\_id);

printf("Enter Grade: ");

scanf("%f", &grade);

// Find student

for (int i = 0; i < num\_students; i++)

{

if (strcmp(students[i].student\_id, student\_id) == 0)

{

// Find course in student's schedule

for (int j = 0; j < students[i].num\_courses; j++)

{

if (strcmp(students[i].courses\_enrolled[j].course\_id, course\_id) == 0)

{

students[i].grades[j] = grade;

printf("Grade updated successfully!\n");

return;

}

}

}

}

printf("Invalid Student ID or Course ID!\n");

}

void handle\_attendance(Faculty faculty)

{

printf("\n--- Handle Attendance ---\n");

char student\_id[20];

printf("Enter Student ID to mark attendance: ");

scanf("%s", student\_id);

int student\_index = -1;

for (int i = 0; i < num\_students; i++)

{

if (strcmp(students[i].student\_id, student\_id) == 0)

{

student\_index = i;

break;

}

}

if (student\_index != -1)

{

printf("Attendance marked for %s.\n", students[student\_index].name);

}

else

{

printf("Student not found!\n");

}

}

void admin\_manage\_students()

{

printf("\n--- Manage Students ---\n");

int choice;

printf("1. Add Student\n2. Remove Student\nEnter Your Choice: ");

scanf("%d", &choice);

if (choice == 1)

{

if (num\_students < MAX\_STUDENTS)

{

Student new\_student;

printf("Enter Student Name: ");

scanf("%s", new\_student.name);

printf("Enter Student ID: ");

scanf("%s", new\_student.student\_id);

new\_student.num\_courses = 0;

students[num\_students++] = new\_student;

printf("Student added successfully!\n");

}

else

{

printf("Cannot add more students, maximum limit reached!\n");

}

}

else if (choice == 2)

{

char student\_id[20];

printf("Enter Student ID to Remove: ");

scanf("%s", student\_id);

int student\_index = -1;

for (int i = 0; i < num\_students; i++)

{

if (strcmp(students[i].student\_id, student\_id) == 0)

{

student\_index = i;

break;

}

}

if (student\_index != -1)

{

for (int j = student\_index; j < num\_students - 1; j++)

{

students[j] = students[j + 1];

}

num\_students--;

printf("Student removed successfully!\n");

}

else

{

printf("Student not found!\n");

}

}

else

{

printf("Invalid Choice!\n");

}

}

void admin\_manage\_faculty()

{

printf("\n--- Manage Faculty ---\n");

int choice;

printf("1. Add Faculty\n2. Remove Faculty\nEnter Your Choice: ");

scanf("%d", &choice);

if (choice == 1)

{

if (num\_faculty < MAX\_FACULTY)

{

char name[50], faculty\_id[20];

printf("Enter Faculty Name: ");

scanf("%s", name);

printf("Enter Faculty ID: ");

scanf("%s", faculty\_id);

strcpy(faculty[num\_faculty].name, name);

strcpy(faculty[num\_faculty].faculty\_id, faculty\_id);

faculty[num\_faculty].num\_courses = 0;

num\_faculty++;

printf("Faculty added successfully!\n");

}

else

{

printf("Cannot add more faculty, limit reached!\n");

}

}

else if (choice == 2)

{

char faculty\_id[20];

printf("Enter Faculty ID to Remove: ");

scanf("%s", faculty\_id);

int faculty\_index = -1;

for (int i = 0; i < num\_faculty; i++)

{

if (strcmp(faculty[i].faculty\_id, faculty\_id) == 0)

{

faculty\_index = i;

break;

}

}

if (faculty\_index != -1)

{

for (int i = faculty\_index; i < num\_faculty - 1; i++)

{

faculty[i] = faculty[i + 1];

}

num\_faculty--;

printf("Faculty removed successfully!\n");

}

else

{

printf("Faculty not found!\n");

}

}

else

{

printf("Invalid Choice!\n");

}

}

void admin\_manage\_courses()

{

printf("\n--- Manage Courses ---\n");

int choice;

printf("1. Add Course\n2. Remove Course\nEnter Your Choice: ");

scanf("%d", &choice);

if (choice == 1)

{

if (num\_courses < MAX\_COURSES)

{

Course new\_course;

printf("Enter Course Name: ");

scanf("%s", new\_course.name);

printf("Enter Course ID: ");

scanf("%s", new\_course.course\_id);

printf("Enter Course Credits: ");

scanf("%d", &new\_course.credits);

courses[num\_courses++] = new\_course;

printf("Course added successfully!\n");

}

else

{

printf("Cannot add more courses, maximum limit reached!\n");

}

}

else if (choice == 2)

{

char course\_id[20];

printf("Enter Course ID to Remove: ");

scanf("%s", course\_id);

int course\_index = -1;

for (int i = 0; i < num\_courses; i++)

{

if (strcmp(courses[i].course\_id, course\_id) == 0)

{

course\_index = i;

break;

}

}

if (course\_index != -1)

{

for (int j = course\_index; j < num\_courses - 1; j++)

{

courses[j] = courses[j + 1];

}

num\_courses--;

printf("Course removed successfully!\n");

}

else

{

printf("Course not found!\n");

}

}

else

{

printf("Invalid Choice!\n");

}

}

void view\_faculties\_and\_students(Student students[], int num\_students, Faculty faculty[], int num\_faculty)

{

printf("\n--- List of Faculties ---\n");

for (int i = 0; i < num\_faculty; i++)

{

printf("Faculty Name: %s, Faculty ID: %s\n", faculty[i].name, faculty[i].faculty\_id);

}

printf("\n--- List of Students ---\n");

for (int i = 0; i < num\_students; i++)

{

printf("Student Name: %s, Student ID: %s\n", students[i].name, students[i].student\_id);

}

}