// SPDX-License-Identifier: MIT

pragma solidity ^0.8.0;

contract Student {

// Define a structure to store student information

struct StudentInfo {

uint id;

string name;

uint age;

string course;

}

// Array to store students

StudentInfo[] public students;

// Mapping to check if student exists by ID

mapping(uint => bool) public studentExists;

// Event to log student addition

event StudentAdded(uint id, string name, uint age, string course);

// Function to add a new student

function addStudent(uint \_id, string memory \_name, uint \_age, string memory \_course) public {

require(!studentExists[\_id], "Student with this ID already exists.");

// Add the new student to the array

students.push(StudentInfo(\_id, \_name, \_age, \_course));

// Mark student ID as existing

studentExists[\_id] = true;

// Emit event

emit StudentAdded(\_id, \_name, \_age, \_course);

}

// Function to get the total number of students

function getTotalStudents() public view returns (uint) {

return students.length;

}

// Receive function to handle incoming Ether

receive() external payable {}

// Fallback function to handle ether sent to the contract with data

fallback() external payable {}

// Function to get the balance of Ether in the contract

function getContractBalance() public view returns (uint) {

return address(this).balance;

}

}