// SPDX-License-Identifier: MIT

pragma solidity ^0.8.0;

contract StudentData {

struct Student {

uint256 id;

string name;

uint256 age;

}

Student[] public students;

address public owner;

constructor() {

owner = msg.sender;

}

modifier onlyOwner() {

require(msg.sender == owner, "Only the owner can perform this action");

\_;

}

function addStudent(uint256 \_id, string memory \_name, uint256 \_age) public onlyOwner {

Student memory newStudent = Student(\_id, \_name, \_age);

students.push(newStudent);

}

function getStudent(uint256 \_index) public view returns (uint256, string memory, uint256) {

require(\_index < students.length, "Student with this index does not exist");

Student memory student = students[\_index];

return (student.id, student.name, student.age);

}

function getStudentCount() public view returns (uint256) {

return students.length;

}

// Fallback function to accept Ether

receive() external payable {

// This function can be used to receive Ether payments

}

}