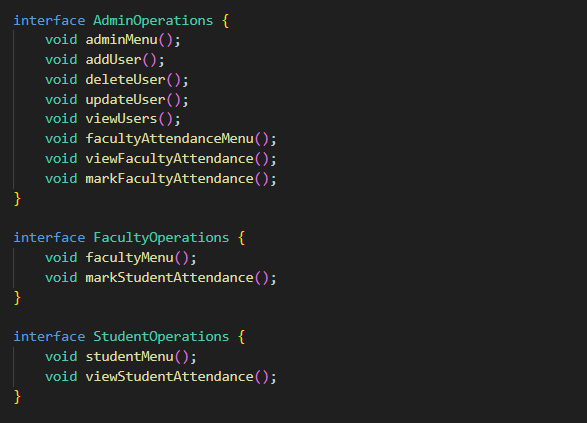
**HASEEB TARIQ**

**232506**

**OOP PROJECT**

**Screenshots of Interfaces:**



**Problems before this code existed: (:**

**Forgetful Faculty**: Ever had a teacher who couldn't remember if you were present or not? This code fixes that! Now, faculty can mark attendance with a few keystrokes, saving them from embarrassing "Were you here yesterday?" moments.

**Vanishing Students:** Ever had your name mysteriously disappear from the attendance list? No more disappearing acts! With this code, students' names and IDs stay put, ensuring they're never accidentally deleted from the system.

**Admin Anarchy:** Tired of chaotic admin procedures where no one knows who's who? Fear not! This code brings order to the admin chaos, allowing admins to add, update, and delete users with ease, restoring peace to the kingdom of user management.

**Menu Mayhem:** Ever felt lost in a sea of options, drowning in a menu with no lifeline? Not anymore! This code offers a life raft in the form of a clear, concise menu, guiding users through the turbulent waters of system interaction.

**Input Insanity:** Fed up with systems that accept gibberish and emojis as input? Say goodbye to input insanity! This code puts an end to nonsensical inputs, ensuring only valid integers pass through its gates, protecting the system from the wild whims of users.

**Proposed Solution:**

**Structured User Management:** The code organizes user data for admins, students, and faculty into arrays, ensuring easy access and manipulation.

**User-Friendly Interfaces:** By utilizing interfaces, it provides clear blueprints for user interactions, making it simpler to understand and maintain.

**Menu-driven Navigation:** The menu-driven approach offers a clear path for users to navigate through various functionalities, reducing confusion and ensuring efficient usage.

**Input Validation:** It incorporates input validation to ensure that only valid integers are accepted, preventing errors caused by incorrect inputs.

**Centralized Attendance Tracking:** With functions for marking and viewing attendance, it centralizes attendance tracking, eliminating the need for separate manual records and reducing the likelihood of errors.

**Modular Design:** The code is designed in a modular manner, allowing for easier maintenance, scalability, and future enhancements.

**Modules and functionality details**

**User Management Module:**

**Admin Operations:**

1. Add Admin: Allows adding new admin users with name, email, and ID.
2. Delete Admin: Enables removing admin users based on their ID.
3. Update Admin: Provides functionality to update existing admin user details.
4. View Admins: Displays a list of all admin users with their details.

**Student Operations:**

1. Add Student: Allows adding new student users with name and ID.
2. Delete Student: Enables removing student users based on their ID.
3. Update Student: Provides functionality to update existing student user details.
4. View Students: Displays a list of all student users with their details.

**Faculty Operations:**

1. Add Faculty: Allows adding new faculty members with name and ID.
2. Delete Faculty: Enables removing faculty members based on their ID.
3. Update Faculty: Provides functionality to update existing faculty member details.
4. View Faculty: Displays a list of all faculty members with their details.

**Attendance Tracking Module:**

**Faculty Attendance:**

1. Mark Faculty Attendance: Allows faculty members to mark their attendance.
2. View Faculty Attendance: Displays the attendance status of all faculty members.

**Student Attendance:**

1. Mark Student Attendance: Allows faculty members to mark the attendance of students.
2. View Student Attendance: Displays the attendance status of individual students.

**User Interface Module:**

1. Menu Navigation: Provides interactive menus for admins, faculty, and students to access various functionalities easily.
2. Input Validation: Ensures that only valid integer inputs are accepted, reducing the risk of errors during user interactions.

**User Classes:**

1. Admin Class: Manages admin-specific operations and interactions.
2. Faculty Class: Handles functionalities specific to faculty members.
3. Student Class: Deals with functionalities tailored for students.

**Data Storage:**

1. Utilizes arrays to store user details and attendance records, ensuring easy access and manipulation of data.

**Input Handling:**

Utilizes a scanner object for user input, providing a seamless interface for interacting with the application.