Student Attendance Tracker Project

This is the Java code for the 'Student Attendance Tracker' project.

import java.io.\*;  
import java.util.\*;  
  
class User {  
 private String username;  
 private String password;  
 public User(String username, String password) {  
 this.username = username;  
 this.password = password;  
 }  
 public String getUsername() { return username; }  
 @Override  
 public String toString() {  
 return username + "," + password;  
 }  
 public static User fromString(String data) {  
 String[] parts = data.split(",");  
 return new User(parts[0], parts[1]);  
 }  
}  
  
class Student {  
 private String id;  
 private String name;  
 public Student(String id, String name) {  
 this.id = id;  
 this.name = name;  
 }  
 public String getId() { return id; }  
 public String getName() { return name; }  
 @Override  
 public String toString() {  
 return id + "," + name;  
 }  
 public static Student fromString(String data) {  
 String[] parts = data.split(",");  
 return new Student(parts[0], parts[1]);  
 }  
}  
  
class AttendanceRecord {  
 private String studentId;  
 private String date;  
 private boolean present;  
 public AttendanceRecord(String studentId, String date, boolean present) {  
 this.studentId = studentId;  
 this.date = date;  
 this.present = present;  
 }  
 @Override  
 public String toString() {  
 return studentId + "," + date + "," + present;  
 }  
 public static AttendanceRecord fromString(String data) {  
 String[] parts = data.split(",");  
 return new AttendanceRecord(parts[0], parts[1], Boolean.parseBoolean(parts[2]));  
 }  
 public String getStudentId() { return studentId; }  
 public String getDate() { return date; }  
 public boolean isPresent() { return present; }  
}  
  
class AttendanceManager {  
 private static final String STUDENT\_FILE = "data/students.txt";  
 private static final String ATTENDANCE\_FILE = "data/attendance.txt";  
 private static final String USER\_FILE = "data/users.txt";  
  
 public AttendanceManager() {  
 new File("data").mkdir();   
 try {  
 new File(STUDENT\_FILE).createNewFile();  
 new File(ATTENDANCE\_FILE).createNewFile();  
 new File(USER\_FILE).createNewFile();  
 } catch (IOException e) {  
 System.out.println("Error initializing files: " + e.getMessage());  
 }  
 }  
  
 public void addUser(User user) throws IOException {  
 try (FileWriter fw = new FileWriter(USER\_FILE, true)) {  
 fw.write(user.toString() + "\n");  
 }  
 }  
  
 public boolean authenticate(String username, String password) throws IOException {  
 try (BufferedReader br = new BufferedReader(new FileReader(USER\_FILE))) {  
 String line;  
 while ((line = br.readLine()) != null) {  
 if (!line.isEmpty()) {  
 User user = User.fromString(line);  
 if (user.getUsername().equals(username) && user.toString().equals(username + "," + password)) {  
 return true;  
 }  
 }  
 }  
 }  
 return false;  
 }  
  
 public void addStudent(Student student) throws IOException {  
 try (FileWriter fw = new FileWriter(STUDENT\_FILE, true)) {  
 fw.write(student.toString() + "\n");  
 }  
 }  
  
 public List<Student> getAllStudents() throws IOException {  
 List<Student> students = new ArrayList<>();  
 try (BufferedReader br = new BufferedReader(new FileReader(STUDENT\_FILE))) {  
 String line;  
 while ((line = br.readLine()) != null) {  
 if (!line.isEmpty()) {  
 students.add(Student.fromString(line));  
 }  
 }  
 }  
 return students;  
 }  
  
 public void markAttendance(String studentId, String date, boolean present) throws IOException {  
 try (FileWriter fw = new FileWriter(ATTENDANCE\_FILE, true)) {  
 fw.write(new AttendanceRecord(studentId, date, present).toString() + "\n");  
 }  
 }  
  
 public List<AttendanceRecord> getAttendanceRecords() throws IOException {  
 List<AttendanceRecord> records = new ArrayList<>();  
 try (BufferedReader br = new BufferedReader(new FileReader(ATTENDANCE\_FILE))) {  
 String line;  
 while ((line = br.readLine()) != null) {  
 if (!line.isEmpty()) {  
 records.add(AttendanceRecord.fromString(line));  
 }  
 }  
 }  
 return records;  
 }  
}  
  
public class Main {  
 public static void main(String[] args) {  
 AttendanceManager manager = new AttendanceManager();  
 Scanner scanner = new Scanner(System.in);  
 System.out.print("Register new user? (y/n): ");  
 if (scanner.nextLine().equalsIgnoreCase("y")) {  
 try {  
 System.out.print("New username: ");  
 String newUser = scanner.nextLine();  
 System.out.print("New password: ");  
 String newPass = scanner.nextLine();  
 manager.addUser(new User(newUser, newPass));  
 System.out.println("✅ User registered.");  
 } catch (IOException e) {  
 System.out.println("⚠ Error during registration: " + e.getMessage());  
 }  
 }  
  
 System.out.println("\n--- Login Required ---");  
 System.out.print("Username: ");  
 String username = scanner.nextLine();  
 System.out.print("Password: ");  
 String password = scanner.nextLine();  
 try {  
 if (!manager.authenticate(username, password)) {  
 System.out.println("❌ Invalid credentials. Exiting.");  
 return;  
 } else {  
 System.out.println("✅ Login successful!");  
 }  
 } catch (IOException e) {  
 System.out.println("⚠ Error reading user file: " + e.getMessage());  
 return;  
 }  
  
 while (true) {  
 System.out.println("\n--- Student Attendance Tracker ---");  
 System.out.println("1. Add Student");  
 System.out.println("2. Mark Attendance");  
 System.out.println("3. View Attendance");  
 System.out.println("4. Exit");  
 System.out.print("Choose option: ");  
 int choice;  
 try {  
 choice = Integer.parseInt(scanner.nextLine());  
 } catch (Exception e) {  
 System.out.println("❌ Invalid input.");  
 continue;  
 }  
 try {  
 switch (choice) {  
 case 1:  
 System.out.print("Enter student ID: ");  
 String id = scanner.nextLine();  
 System.out.print("Enter student name: ");  
 String name = scanner.nextLine();  
 manager.addStudent(new Student(id, name));  
 System.out.println("✅ Student added.");  
 break;  
 case 2:  
 List<Student> students = manager.getAllStudents();  
 if (students.isEmpty()) {  
 System.out.println("⚠ No students found.");  
 break;  
 }  
 System.out.print("Enter date (YYYY-MM-DD): ");  
 String date = scanner.nextLine();  
 for (Student s : students) {  
 System.out.print("Is " + s.getName() + " present? (y/n): ");  
 String ans = scanner.nextLine();  
 manager.markAttendance(s.getId(), date, ans.equalsIgnoreCase("y"));  
 }  
 System.out.println("✅ Attendance marked.");  
 break;  
 case 3:  
 List<AttendanceRecord> records = manager.getAttendanceRecords();  
 if (records.isEmpty()) {  
 System.out.println("⚠ No attendance records found.");  
 } else {  
 List<Student> allStudents = manager.getAllStudents();  
 Map<String, String> studentMap = new HashMap<>();  
 for (Student s : allStudents) {  
 studentMap.put(s.getId(), s.getName());  
 }  
  
 System.out.println("\n📋 Attendance Records:");  
 for (AttendanceRecord r : records) {  
 String studentName = studentMap.getOrDefault(r.getStudentId(), "Unknown");  
 System.out.printf("Name: %s | Date: %s | Present: %s\n",  
 studentName, r.getDate(), r.isPresent() ? "Yes" : "No");  
 }  
 }  
 break;  
 case 4:  
 System.out.println("👋 Exiting. Bye!");  
 return;  
 default:  
 System.out.println("❌ Invalid option. Try again.");  
 }  
 } catch (IOException e) {  
 System.out.println("⚠ Error: " + e.getMessage());  
 }  
 }  
 }  
}