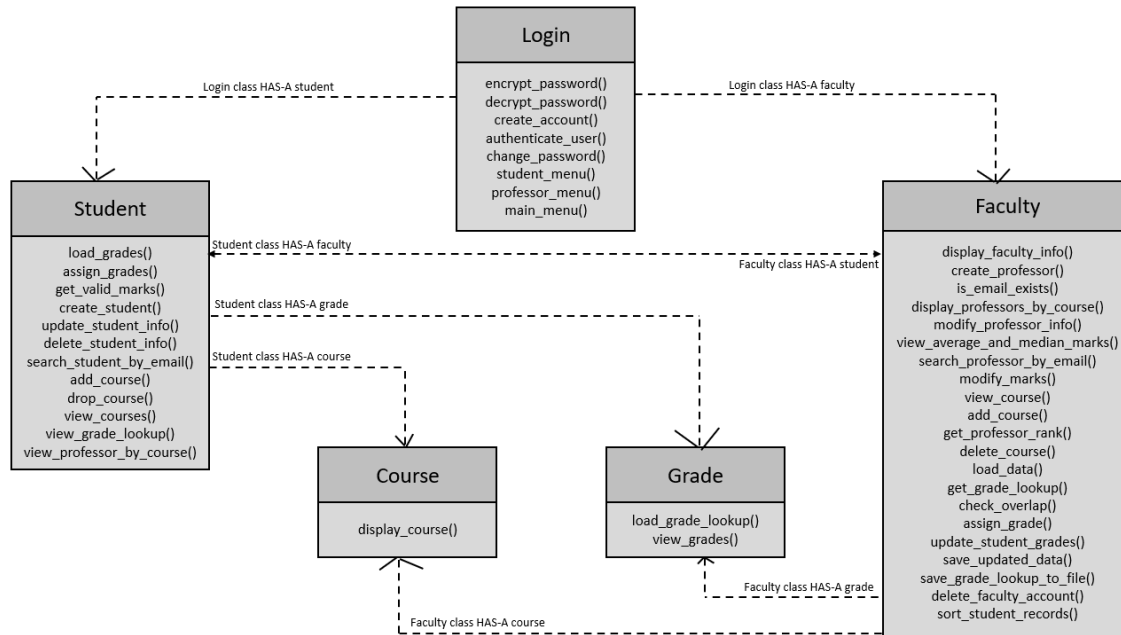


Check My Grade Application

Objective: The objective of this lab is to create an application “Check My Grade” which allow users to login and perform various functions like add/delete course, manipulate marks, check grades, etc.

Class Relationship Diagram



GitHub Link

The GitHub repository contains

- student_records.csv
- login.csv
- faculty_data.csv
- course_records.csv
- grades.csv
- login_system.py
- student_system.py
- course_system.py
- grade_system.py
- faculty_system.py
- main.py
- Unit_test.py
- Goel_Lab1_Report.pdf

GitHub repository link - https://github.com/GoelAnshika99/DATA200_Lab1

Note: The credentials for all student and professor are stored in *login.csv*. The password has been set to ‘welcome’ for every user which is encrypted and stored in *login.csv* as ‘[jqhtrj]’. To access data for any existing student/professor input the email as username and password as ‘welcome’.

Detailed Overview

The application consists of four different classes.

- **Login**

Class login adds functionality that allows user to successfully login and signup users. The member functions included in this class are given below:

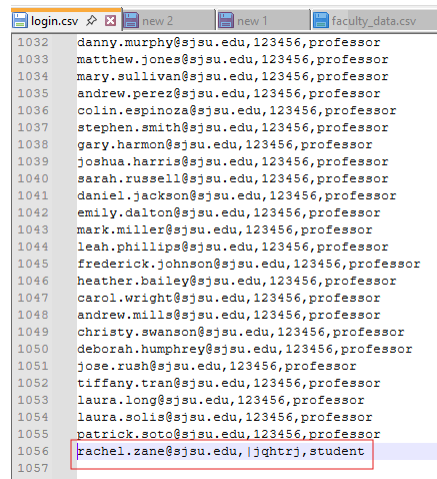
- `encrypt_password()` – encrypts the password before storing in *login.csv*

```
def encrypt_password(self, password):  
    encrypted = ''.join(chr((ord(char) + self.shift_value) % 256) for char in password)  
    return encrypted
```

- `decrypt_password()` – decrypts the password from *login.csv*

```
def decrypt_password(self, encrypted_password):  
    decrypted = ''.join(chr((ord(char) - self.shift_value) % 256) for char in encrypted_password)  
    return decrypted
```

- `create_account()` – creates an account for new user and stores the details in *login.csv*.



ID	email	ID	password	role
1032	danny.murphy@sjsu.edu	123456	professor	
1033	matthew.jones@sjsu.edu	123456	professor	
1034	mary.sullivan@sjsu.edu	123456	professor	
1035	andrew.perez@sjsu.edu	123456	professor	
1036	colin.espinoza@sjsu.edu	123456	professor	
1037	stephen.smith@sjsu.edu	123456	professor	
1038	gary.harmon@sjsu.edu	123456	professor	
1039	joshua.harris@sjsu.edu	123456	professor	
1040	sarah.russell@sjsu.edu	123456	professor	
1041	daniel.jackson@sjsu.edu	123456	professor	
1042	emily.dalton@sjsu.edu	123456	professor	
1043	mark.miller@sjsu.edu	123456	professor	
1044	leah.phillips@sjsu.edu	123456	professor	
1045	frederick.johnson@sjsu.edu	123456	professor	
1046	heather.bailey@sjsu.edu	123456	professor	
1047	carol.wright@sjsu.edu	123456	professor	
1048	andrew.mills@sjsu.edu	123456	professor	
1049	christy.swanson@sjsu.edu	123456	professor	
1050	deborah.humphrey@sjsu.edu	123456	professor	
1051	jose.rush@sjsu.edu	123456	professor	
1052	tiffany.tran@sjsu.edu	123456	professor	
1053	laura.long@sjsu.edu	123456	professor	
1054	laura.solis@sjsu.edu	123456	professor	
1055	patrick.soto@sjsu.edu	123456	professor	
1056	rachel.zane@sjsu.edu	jqhtrj	student	
1057				

New account created in login.csv with encrypted password

```

login_system.py > Login > create_account
4 class Login:
14 def create_account(self, role):
15     print(f"Creating a new {role} account.....")
16     while True:
17         email = input("Enter your email: ")
18         email_exists = False
19         with open(self.csv_file, mode = 'r') as file:
20             reader = csv.reader(file)
21             for row in reader:
22                 stored_email, _, stored_role = row
23                 if stored_email == email:
24                     email_exists = True
25                     break
26         if email_exists:
27             print("Email already exists. Please enter a different email.")
28         else:
29             password = input("Enter your password: ")
30             encrypted_password = self.encrypt_password(password)
31             with open(self.csv_file, mode = 'a') as file:
32                 writer = csv.writer(file)
33                 writer.writerow([email, encrypted_password, role])
34             print(f"Account created successfully for {email} as a {role}.")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\aradh\Desktop\DATA 200 Python Programming\Lab1> & C:/Users/aradh/AppData/Local/Programs/Python/Python38-32/Python.exe C:/Users/aradh/Desktop/DATA 200 Python Programming/Lab1/main.py

-----Main Menu-----
1. Student
2. Professor
3. Exit
Please choose an option (1/2/3): 1

-----Student Menu-----
1. New User
2. Existing User
3. Return to main menu
Please choose an option (1/2/3): 1
Creating a new student account.....
Enter your email: rachel.zane@sjsu.edu
Enter your password:
Account created successfully for rachel.zane@sjsu.edu as a student.

```

Console output

- `authenticate_user()` – authenticates the existing user. Decrypts the password stored in `login.csv` by using `decrypt_password()` and then matches with the user input. If the password is correct, it allows the user to further access the “Check My Grade” application.

```

login_system.py X student_system.py main.py course_system.py grade_system.py
login_system.py > Login > create_account
4 class Login:
36 def authenticate_user(self, role):
37     print(f"Please log in as a {role}.")
38     email = input("Enter your email: ")
39     password = getpass.getpass("Enter your password: ")
40     encrypted_input_password = self.encrypt_password(password)
41     with open(self.csv_file, mode = 'r') as file:
42         reader = csv.reader(file)
43         for row in reader:
44             stored_email, stored_encrypted_password, stored_role = row
45             if stored_role == role and stored_email == email:
46                 if stored_encrypted_password == encrypted_input_password:
47                     print(f"Login successful! Welcome back, {email} ({stored_role}).")
48                     return True
49             print("Login failed. Incorrect email or password.")
50     return False
51 def change_password(self, role, email, new_password):
52     print(f"Changing password for {email} as a {role}.")
53     old_password = getpass.getpass("Enter your old password: ")
54     encrypted_old_password = self.encrypt_password(old_password)
55     with open(self.csv_file, mode = 'r') as file:
56         reader = csv.reader(file)
57         for row in reader:
58             stored_email, stored_encrypted_password, stored_role = row
59             if stored_role == role and stored_email == email:
60                 if stored_encrypted_password == encrypted_old_password:
61                     new_encrypted_password = self.encrypt_password(new_password)
62                     with open(self.csv_file, mode = 'w') as file:
63                         writer = csv.writer(file)
64                         writer.writerow([email, new_encrypted_password, role])
65                     print(f"Password changed successfully for {email} as a {role}.")
66                     return True
67             print("Login failed. Incorrect email or password.")
68     return False

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

-----Student Menu-----
1. New User
2. Existing User
3. Return to main menu
Please choose an option (1/2/3): 2
Please log in as a student.
Enter your email: rachel.zane@sjsu.edu
Enter your password:
Login successful! Welcome back, rachel.zane@sjsu.edu (student).

```

- `change_password()` – allows user to change password

login.csv	new 2	new 1	faculty_data.csv
1032	danny.murphy@sjsu.edu,123456,professor		
1033	matthew.jones@sjsu.edu,123456,professor		
1034	mary.sullivan@sjsu.edu,123456,professor		
1035	andrew.perez@sjsu.edu,123456,professor		
1036	colin.espinoza@sjsu.edu,123456,professor		
1037	stephen.smith@sjsu.edu,123456,professor		
1038	gary.harmon@sjsu.edu,123456,professor		
1039	joshua.harris@sjsu.edu,123456,professor		
1040	sarah.russell@sjsu.edu,123456,professor		
1041	daniel.jackson@sjsu.edu,123456,professor		
1042	emily.dalton@sjsu.edu,123456,professor		
1043	mark.miller@sjsu.edu,123456,professor		
1044	leah.phillips@sjsu.edu,123456,professor		
1045	frederick.johnson@sjsu.edu,123456,professor		
1046	heather.bailey@sjsu.edu,123456,professor		
1047	carol.wright@sjsu.edu,123456,professor		
1048	andrew.mills@sjsu.edu,123456,professor		
1049	christy.swanson@sjsu.edu,123456,professor		
1050	deborah.humphrey@sjsu.edu,123456,professor		
1051	jose.rush@sjsu.edu,123456,professor		
1052	tiffany.tran@sjsu.edu,123456,professor		
1053	laura.long@sjsu.edu,123456,professor		
1054	laura.solis@sjsu.edu,123456,professor		
1055	patrick.soto@sjsu.edu,123456,professor		
1056	rachel.zane@sjsu.edu, jqhtrj6,student		
1057			

Password changed for user 'rachel.zane@sjsu.edu'

```

login_system.py X student_system.py main.py course_system.py grade_system.py
login_system.py > Login > create_account
4 class Login:
51 def change_password(self, email, role):
52     old_password = getpass.getpass("Enter your old password: ")
53     encrypted_old_password = self.encrypt_password(old_password)
54     with open(self.csv_file, mode = 'r') as file:
55         reader = csv.reader(file)
56         rows = list(reader)
57         found = False
58         for row in rows:
59             if row[0] == email and row[2] == role and row[1] == encrypted_old_password:
60                 found = True
61                 new_password = getpass.getpass("Enter your new password: ")
62                 encrypted_new_password = self.encrypt_password(new_password)
63                 row[1] = encrypted_new_password
64                 break
65         if found:
66             with open(self.csv_file, mode = 'w', newline = '') as file:
67                 writer = csv.writer(file)
68                 writer.writerows(rows)
69
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
-----Sub Student Menu-----
1. View your records
2. Update your information
3. Delete your account
4. Search student by email
4. Search student by email
5. Add course
6. Delete course
7. View Course List
8. View Grade Lookup
9. View professor by course
10. Change Password
11. Return to main menu
Please choose an option (1/2/3/4/5/6/7/8/9/10/11): 10
Enter your email: rachel.zane@sjsu.edu
Enter your old password:
Enter your new password:
Password changed successfully for rachel.zane@sjsu.edu.

```

Console Output

- student_menu() – displays menu to the user if the user is successfully authenticated as a student. The menu given below represents functionality that a student can perform while accessing “Check My Grade” application.
 - View student information
 - Update information like address and phone number
 - Delete student account

- Search student using email
- Add course to a student record
- Drop course from a student record
- View list of courses available
- View grade lookup
- View list of professors grouped by course ID
- Change password for their account

The screenshot shows a code editor with several tabs: login_system.py, student_system.py, main.py, course_system.py, and grade_system.py. The 'login_system.py' tab is active, showing the 'Login' class and the 'student_menu' method. The method is a loop that displays a menu of 11 options for a student. The terminal output shows the menu being displayed and the user selecting option 10.

```

login_system.py > Login > create_account
4 class Login:
72 def student_menu(self, student, email):
73     from student_system import Student
74     while True:
75         print("\n-----Sub Student Menu-----")
76         print("1. View your records")
77         print("2. Update your information")
78         print("3. Delete your account")
79         print("4. Search student by email")
80         print("5. Add course")
81         print("6. Delete course")
82         print("7. View Course List")
83         print("8. View Grade Lookup")
84         print("9. View professor by course")
85         print("10. Change Password")
86         print("11. Return to main menu")
87         choice = input("Please choose an option (1/2/3/4/5/6/7/8/9/10/11): ")
88         if choice == '1':
89             email = input("Enter your email to view your records: ")
90             student.view_student_records(email)
91         elif choice == '2':

```

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

-----Sub Student Menu-----
1. View your records
2. Update your information
3. Delete your account
4. Search student by email
4. Search student by email
5. Add course
6. Delete course
7. View Course List
8. View Grade Lookup
9. View professor by course
10. Change Password
11. Return to main menu
Please choose an option (1/2/3/4/5/6/7/8/9/10/11): 10

```

- professor_menu() – displays menu to the user if the user is successfully authenticated as a professor. The menu given below represents functionality that a professor can perform while accessing “Check My Grade” application.
 - View professor information
 - View list of professors grouped by course ID
 - Update information like address and phone number
 - View course average and median marks to analyze course performance (output is sorted by average marks in decreasing order)
 - Sort student records (either by email or marks)
 - Search records for a particular student
 - Search records for a particular professor
 - Modify marks of a student
 - Delete a student account (only the head of department is permitted to delete student record)
 - View list of available courses
 - Add a course (only the head of department is permitted to add a new course)
 - Delete a course (only the head of department is permitted to delete an existing course)
 - Modify grade lookup (only the head of department is permitted to update grade lookup)
 - Delete professor account

- Change password

The screenshot shows a code editor with several tabs: login_system.py, student_system.py, main.py, course_system.py, and gra. The active tab is login_system.py, which contains the following code:

```

4 class Login:
121     def professor_menu(self, faculty, email):
122         from faculty_system import Faculty
123         from student_system import Student
124         while True:
125             print("\n-----Sub Professor Menu-----")
126             print("1. View your details")
127             print("2. View professor list by Course ID")
128             print("3. Modify your details")
129             print("4. View Course Performance (sorted by average marks)")
130             print("5. Sort student data by email or marks")
131             print("6. Search student by email")
132             print("7. Search professor by email")
133             print("8. Modify student marks")
134             print("9. Delete student account")
135             print("10. View Course List")
136             print("11. Add Course")
137             print("12. Delete Course")
138             print("13. Modify Grade Lookup")
139             print("14. Delete your account")
140             print("15. Change password")

```

Below the code editor, the terminal window shows the output of the professor_menu method:

```

-----Sub Professor Menu-----
1. View your details
2. View professor list by Course ID
3. Modify your details
4. View Course Performance (sorted by average marks)
5. Sort student data by email or marks
6. Search student by email
7. Search professor by email
8. Modify student marks
9. Delete student account
10. View Course List
11. Add Course
12. Delete Course
13. Modify Grade Lookup
14. Delete your account
15. Change password
16. Return to main menu

```

- main_menu() – displays option to login as a student or a professor
 - Student
 - New User – enables user to sign-up as a student and create username and password
 - Existing User – enables user to login as a student. Authenticates user based on the username and password entered
 - Professor
 - New User – enables user to sign-up as a professor and create username and password
 - Existing User – enables user to login as a professor. Authenticates user based on the username and password entered

```

class Login:
    def main_menu(self):
        from student_system import Student
        from faculty_system import Faculty
        while True:
            print("\n-----Main Menu-----")
            print("1. Student")
            print("2. Professor")
            print("3. Exit")
            choice = input("Please choose an option (1/2/3): ")
            if choice == '1':
                while True:
                    print("\n-----Student Menu-----")
                    print("1. New User")
                    print("2. Existing User")
                    print("3. Return to main menu")
                    student_choice = input("Please choose an option (1/2/3): ")
                    if student_choice == '1':
                        email = self.create_account("Student")
                        student_records = "C:/Users/aradh/Desktop/DATA 200 Python Programming/Lab1/student_records.csv"
                        grade_file = "C:/Users/aradh/Desktop/DATA 200 Python Programming/Lab1/grades.csv"

```

PS C:\Users\aradh\Desktop\DATA 200 Python Programming\Lab1> & C:/Users/aradh/AppData/Local/Programs/Python/Python313/python.exe "ATA 200 Python Programming\Lab1/main.py"

```

-----Main Menu-----
1. Student
2. Professor
3. Exit
Please choose an option (1/2/3): 1
-----Student Menu-----
1. New User
2. Existing User
3. Return to main menu
Please choose an option (1/2/3): 1
Creating a new student account.....
Enter your email: rachel.zane@jsu.edu
Enter your password:

```

• Student

The student class adds functionality to the student profile and allows user to perform multiple actions.

- `load_grades()`: used to read file `grades.csv`

```

class Student:
    def load_grades(self):
        grades = []
        with open(self.grades_file, mode = 'r') as file:
            reader = csv.reader(file)
            next(reader)
            for row in reader:
                try:
                    if len(row) >= 3:
                        grade = row[0]
                        min_marks = int(row[2])
                        max_marks = int(row[1])
                        grades[grade] = (max_marks, min_marks)
                except ValueError as e:
                    print(f"Error processing row {row}: {e}")
        return grades

```

- `assign_grades()`: assign grades to the marks entered by the user based on the grade lookup stored in `grades.csv`

```

class Student:
    def assign_grades(self, marks):
        grades = self.load_grades()
        for grade, (max_marks, min_marks) in grades.items():
            if min_marks <= marks <= max_marks:
                return grade
        return 'F'

```

- `get_valid_marks()`: The “Check My Grade” application assumes that a maximum and minimum marks that a student can score is 100 and 0 respectively. The function `get_valid_marks()` checks if the marks entered by the user lies between 0 and 100.

```

login_system.py student_system.py X main.py course_system.py grade_system.py fa
student_system.py > Student > load_grades
8 class Student:
37     def get_valid_marks(self):
38         while True:
39             try:
40                 marks = int(input("Enter your marks for the course (between 0 and 100): "))
41                 if 0 <= marks <= 100:
42                     return marks
43             except:
44                 print("Error: Marks must be between 0 and 100. Please try again.")
45             except ValueError:
46                 print("Error: Invalid input. Please enter a valid integer between 0 and 100.")

```

- create_student(): Once the user decides to sign-up as a new student the function create_student() creates a new record for the student in *student_records.csv*

```

student_records.csv login.csv new 2 new 1 faculty_data.csv course_records.csv test_course_records.csv test_fa
984 Brianna, Henry, brandy39@example.org, "594 Catherine Way, West Katherineborough, PA 04678", 8466612661, data200, 1, F
985 Jeffrey, Taylor, danielastin@example.com, "20625 Linda Dale Suite 389, Port Tiffanystad, NM 51195", 8918659393, data245, 1, F
986 Jennifer, Jensen, dgilbert@example.com, "108 Jennifer Overpass Apt. 118, West Maryberg, TX 09583", 7756560489, data200, 1, F
987 Brittany, Ibarra, fbennett@example.org, "423 Lisa Falls, West Cassandraview, NV 31087", 8782883500, data200, 1, F
988 Justin, Roach, katherinemcdonald@example.org, "USS Gordon, FPO AF 42920", 9907336479, data220, 1, F
989 Darren, Goodwin, kimberly21@example.com, "07347 Anderson Lane, New Sarah, SC 83235", 8080240461, data228, 1, F
990 Shelia, Bowen, ronald10@example.com, "0395 Stephanie Tunnel Suite 658, Lake Jeffreyburgh, LA 75541", 9758831305, data245, 0, F
991 Mark, Skinner, sschmidt@example.org, "051 Pearson Spur Apt. 656, North Edwardview, TX 66546", 8649397867, data255, 1, F
992 John, Decker, theresavasquez@example.org, "84679 Stephens Center Suite 910, Smithmouth, IN 86618", 9925340453, data200, 1, F
993 Alexis, Valencia, bergion@example.net, "4182 Johnston Camp Apt. 205, Port Tammyland, RI 45635", 6618405205, data245, 0, F
994 Laura, Daniels, brian44@example.com, "42380 George Throughway Apt. 098, North Kim, CA 22320", 8687820873, data230, 0, F
995 Billy, Tucker, corysantana@example.com, "68570 Hogan Fork Apt. 117, Lake Joseph, NJ 68289", 6191970140, data226, 0, F
996 Nancy, Jones, dakotawelch@example.net, "6330 Faith Parks Suite 115, Lake Zacharyhaven, OK 70787", 6378499636, data220, 0, F
997 Mark, Baker, drodgers@example.net, "180 Jessica Parkways, West Kayla, GA 43584", 6749721830, data228, 0, F
998 Alex, Clay, graysean@example.org, "4699 Hampton Valleys Apt. 253, Williamhaven, FW 39382", 8375955826, data245, 0, F
999 Michael, Smith, janetnewman@example.org, "5103 Ronnie Green Suite 455, Clineland, IL 88679", 7587695741, data255, 0, F
1000 David, Hopkins, soniachandler@example.com, "16803 Spencer Estates, South Garytown, MP 06515", 7881817363, data228, 0, F
1001 Steven, Barry, valerierodriguez@example.com, "35671 Pittman River, Lake Jeannville, HI 73261", 7129815440, data201, 0, F
1002 Rachel, Zane, rachel.zane@sjsu.edu, San Jose, 9432600435, data200, 89, A+
1003

```

New student record for 'rachel.zane@sjsu.edu' in student_records.csv

```

login_system.py student_system.py X main.py course_system.py grade_syst
student_system.py > Student > load_grades
8 class Student:
47     def create_student(self, email):
48         course_ids = []
49         try:
50             with open(self.course_file, mode = 'r') as file:
51                 reader = csv.reader(file)
52                 for row in reader:
53                     course_ids.append(row[1])
54             except FileNotFoundError:
55                 print(f"Error: The course file was not found.")
56             return
57         found = False
58         with open(self.student_records, mode = 'r') as file:
59             reader = csv.reader(file)
60             for row in reader:
61                 if row[2] == email:
62                     found = True
63                     print(f"A record with this email ({email}) already exists.")
64                     return
65             if not found:
66                 first_name = input("Enter your first name: ")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Creating a new student account.....
Enter your email: rachel.zane@sjsu.edu
Enter your password:
Account created successfully for rachel.zane@sjsu.edu as a student.
Enter your first name: Rachel
Enter your last name: Zane
Enter your address: San Jose
Enter your phone number: 9432600435
Enter your course id (or 'done' to finish): data200
Enter your marks for the course (between 0 and 100): 89
Enter your course id (or 'done' to finish): done
Student record created for Rachel Zane.

```

Console Output

- view_student_records(): used to view records for a particular student sorted in *student_records.csv*.


```

login_system.py student_system.py X main.py course_system.py
student_system.py > Student > load_grades
8 class Student:
89     def view_student_records(self, email):
90         found = False
91         total_marks = 0
92         num_courses = 0
93         courses = []
94         with open(self.student_records, mode = 'r') as file:
95             reader = csv.reader(file)
96             for row in reader:
97                 if row[2] == email:
98                     if not found:
99                         print("Student Record:")
100                         print(f"Name: {row[0]} {row[1]}")
101                         print(f"Email: {row[2]}")
102                         print(f"Address: {row[3]}")
103                         print(f"Phone Number: {row[4]}")
104                         found = True
105                         course_id = row[5]
106                         marks = int(row[6])
107                         courses.append((course_id, marks, row[7]))
108                         total_marks += marks

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

10. Change Password
11. Return to main menu
Please choose an option (1/2/3/4/5/6/7/8/9/10/11): 1
Enter your email to view your records: rachel.zane@sjsu.edu
Student Record:
Name: Rachel Zane
Email: rachel.zane@sjsu.edu
Address: San Jose
Phone Number: 9432600435
Course ID: data200
Marks: 89
Grade: A+
-----

Total Marks: 89
Average Marks: 89.00

```

- `update_student_info()`: The “Check My Grade” application assumes that once a user has registered their first name, last name, email cannot be modified. However, details like phone number and address can change and thus allow the user to modify these details.

```

student_records.csv login.csv new 2 new 1 faculty_data.csv course_records.csv test_course_records.csv test_fa
984 Brianna, Henry, brandy39@example.org, "594 Catherine Way, West Katherineborough, PA 04678", 8466612661, data200, 1, F
985 Jeffrey, Taylor, danielastin@example.com, "20625 Linda Dale Suite 389, Port Tiffanystad, NM 51195", 8918659393, data245, 1, F
986 Jennifer, Jensen, dgilbert@example.com, "108 Jennifer Overpass Apt. 118, West Maryberg, TX 09583", 7756560489, data200, 1, F
987 Brittany, Ibarra, fbennett@example.org, "423 Lisa Falls, West Cassandraview, NV 31087", 8782883500, data200, 1, F
988 Justin, Roach, katherinemcdonald@example.org, "USS Gordon, FPO AP 42920", 9907336479, data220, 1, F
989 Darren, Goodwin, kimberly21@example.com, "07347 Anderson Lane, New Sarah, SC 83235", 8080240461, data228, 1, F
990 Shelia, Bowen, ronald10@example.com, "0395 Stephanie Tunnel Suite 658, Lake Jeffreyburgh, LA 75541", 9758831305, data202, 1, F
991 Mark, Skinner, sschmidt@example.org, "051 Pearson Spur Apt. 656, North Edwardview, TX 66546", 8649397867, data255, 1, F
992 John, Decker, theresavasquez@example.org, "84679 Stephens Center Suite 910, Smithmouth, IN 86618", 9925340453, data200, 1, F
993 Alexis, Valencia, bergjon@example.net, "4182 Johnston Camp Apt. 205, Port Tammyland, RI 45635", 6618405205, data245, 0, F
994 Laura, Daniels, brian44@example.com, "42380 George Throughway Apt. 098, North Kim, CA 22320", 8687820873, data230, 0, F
995 Billy, Tucker, corysantana@example.com, "68570 Hogan Fork Apt. 117, Lake Joseph, NJ 68289", 6191970140, data226, 0, F
996 Nancy, Jones, dakotawelch@example.net, "6330 Faith Parks Suite 115, Lake Zacharyhaven, OK 70787", 6378499636, data220, 0, F
997 Mark, Baker, drodgers@example.net, "180 Jessica Parkways, West Kayla, GA 43584", 6749721930, data228, 0, F
998 Alex, Clay, graysean@example.org, "4699 Hampton Valleys Apt. 253, Williamshaven, PW 39382", 8375955826, data245, 0, F
999 Michael, Smith, janetnewman@example.org, "5103 Ronnie Green Suite 455, Clineland, IL 88679", 7587695741, data255, 0, F
1000 David, Hopkins, soniachandler@example.com, "16803 Spencer Estates, South Garytown, MP 06515", 7881817363, data228, 0, F
1001 Steven, Barry, valerierodriguez@example.com, "35671 Fittman River, Lake Jeanville, HI 73261", 7129815440, data201, 0, F
1002 Rachel, Zane, rachel.zane@sjsu.edu, Santa Clara, 6501240356, data200, 89, A+
1003

```

Updated record for 'rachel.zane@sjsu.edu' in student_records.csv

```

login_system.py student_system.py X main.py course_system.py grade_system.py faculty_sys
student_system.py > Student > load_grades
8 class Student:
125     def update_student_info(self, email):
126         row = []
127         found = False
128         with open(self.student_records, mode = 'r') as file:
129             reader = csv.reader(file)
130             rows = list(reader)
131         for row in rows:
132             if row[2] == email:
133                 print(f"Found your record: {row}")
134                 row[3] = input(f"Enter new address (current: {row[3]}): ")
135                 row[4] = input(f"Enter new phone number (current: {row[4]}): ")
136                 found = True
137                 break
138         if found:
139             with open(self.student_records, mode = 'w', newline = '') as file:
140                 writer = csv.writer(file)
141                 writer.writerows(rows)
142                 print("Your information has been updated.")
143         else:
144             print("Record not found.")

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

2. Update your information
3. Delete your account
4. Search student by email
5. Add course
6. Delete course
7. View Course List
8. View Grade Lookup
9. View professor by course
10. Change Password
11. Return to main menu
Please choose an option (1/2/3/4/5/6/7/8/9/10/11): 2
Enter your email to update your information: rachel.zane@sjsu.edu
Found your record: ['Rachel', 'Zane', 'rachel.zane@sjsu.edu', 'San Jose', '9432600435', 'data200', '89', 'A+']
Enter new address (current: San Jose): Santa Clara
Enter new phone number (current: 9432600435): 6501240356
Your information has been updated.

```

Console Output

- `delete_student_account()`: deletes the records of a student from `student_records.csv` and login details from `login.csv`

student_records.csv	login.csv	new 2	new 1	faculty_data.csv	course_records.csv	test_course_records.csv	test_facu
984	Brianna, Henry, brandy39@example.org, "594 Catherine Way, West Katherineborough, PA 04678", 8466612661, data200, 1, F						
985	Jeffrey, Taylor, daniel.austin@example.com, "20625 Linda Dale Suite 389, Port Tiffanystad, NM 51195", 8918659393, data245, 1, F						
986	Jennifer, Jensen, dgilbert@example.com, "108 Jennifer Overpass Apt. 118, West Maryberg, TX 09583", 7756560489, data200, 1, F						
987	Brittany, Ibarra, fbennett@example.org, "423 Lisa Falls, West Cassandraview, NV 31087", 8782883500, data200, 1, F						
988	Justin, Roach, katherinemcdonald@example.org, "USS Gordon, FPO AP 42920", 9907336479, data220, 1, F						
989	Darren, Goodwin, kimberly21@example.com, "07347 Anderson Lane, New Sarah, SC 83235", 8080240461, data228, 1, F						
990	Shelia, Bowen, ronald10@example.com, "0395 Stephanie Tunnel Suite 658, Lake Jeffreyburgh, LA 75541", 9758831305, data202, 1, F						
991	Mark, Skinner, sschmidt@example.org, "051 Pearson Spur Apt. 656, North Edwardview, TX 66546", 8649397867, data255, 1, F						
992	John, Decker, theresavasquez@example.org, "84679 Stephens Center Suite 910, Smithmouth, IN 86618", 9925340453, data200, 1, F						
993	Alexis, Valencia, bergjon@example.net, "4182 Johnston Camp Apt. 205, Port Tammyland, RI 45635", 6618405205, data245, 0, F						
994	Laura, Daniels, brian44@example.com, "42390 George Thoroughway Apt. 098, North Kim, CA 22320", 8687820879, data230, 0, F						
995	Billy, Tucker, corysantana@example.com, "68570 Hogan Fork Apt. 117, Lake Joseph, NJ 68289", 6191970140, data226, 0, F						
996	Nancy, Jones, dakotawelch@example.net, "6330 Faith Parks Suite 115, Lake Zacharyhaven, OK 70787", 6378499636, data220, 0, F						
997	Mark, Baker, drodgers@example.net, "180 Jessica Parkways, West Kayla, GA 43584", 6749721930, data228, 0, F						
998	Alex, Clay, graysean@example.org, "4699 Hampton Valleys Apt. 253, Williamsaven, PW 39382", 8375955826, data245, 0, F						
999	Michael, Smith, janetnewman@example.org, "5103 Ronnie Green Suite 455, Clineand, IL 88679", 7587695741, data255, 0, F						
1000	David, Hopkins, soniachandler@example.com, "16803 Spencer Estates, South Garytown, MP 06515", 7881817363, data228, 0, F						
1001	Steven, Barry, valerierodriguez@example.com, "35671 Pittman River, Lake Jeanville, HI 73261", 7129815440, data201, 0, F						
1002							

No records of 'rachel.zane@sjsu.edu' in student_records.csv

login.csv	student_records.csv	new 2	new
1032	danny.murphy@sjsu.edu,123456,professor		
1033	matthew.jones@sjsu.edu,123456,professor		
1034	mary.sullivan@sjsu.edu,123456,professor		
1035	andrew.perez@sjsu.edu,123456,professor		
1036	colin.espinoza@sjsu.edu,123456,professor		
1037	stephen.smith@sjsu.edu,123456,professor		
1038	gary.harmon@sjsu.edu,123456,professor		
1039	joshua.harris@sjsu.edu,123456,professor		
1040	sarah.russell@sjsu.edu,123456,professor		
1041	daniel.jackson@sjsu.edu,123456,professor		
1042	emily.dalton@sjsu.edu,123456,professor		
1043	mark.miller@sjsu.edu,123456,professor		
1044	leah.phillips@sjsu.edu,123456,professor		
1045	frederick.johnson@sjsu.edu,123456,professor		
1046	heather.bailey@sjsu.edu,123456,professor		
1047	carol.wright@sjsu.edu,123456,professor		
1048	andrew.mills@sjsu.edu,123456,professor		
1049	christy.swanson@sjsu.edu,123456,professor		
1050	deborah.humphrey@sjsu.edu,123456,professor		
1051	jose.rush@sjsu.edu,123456,professor		
1052	tiffany.tran@sjsu.edu,123456,professor		
1053	laura.long@sjsu.edu,123456,professor		
1054	laura.solis@sjsu.edu,123456,professor		
1055	patrick.soto@sjsu.edu,123456,professor		
1056	sheldon.cooper@sjsu.edu,1jqhtrj,professor		
1057			

No record of 'rachel.zane@sjsu.edu' in login.csv

```

login_system.py student_system.py main.py course_system.py grade_system.py fac
student_system.py > Student > load_grades
8 class Student:
145 def delete_student_account(self, email):
146     rows = []
147     with open(self.student_records, mode = 'r') as file:
148         reader = csv.reader(file)
149         rows = list(reader)
150         initial_row_count = len(rows)
151         rows = [row for row in rows if row[2] != email]
152         if len(rows) < initial_row_count:
153             with open(self.student_records, mode = 'w', newline = '') as file:
154                 writer = csv.writer(file)
155                 writer.writerows(rows)
156             print(f"Student account with email {email} has been deleted from student records.")
157         else:
158             print(f"No student records found for email {email} in student records.")
159         login_rows = []
160         with open(self.login_file, mode = 'r') as file:
161             reader = csv.reader(file)
162             login_rows = list(reader)
163             login_row_count = len(login_rows)
164             login_rows = [row for row in login_rows if row[0] != email or row[2] != "student"]

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
-----Sub Student Menu-----
1. View your records
2. Update your information
3. Delete your account
4. Search student by email
5. Add course
6. Delete course
7. View Course List
8. View Grade Lookup
9. View professor by course
10. Change Password
11. Return to main menu
Please choose an option (1/2/3/4/5/6/7/8/9/10/11): 3
Enter your email to delete your account: rachel.zane@sjsu.edu
Student account with email rachel.zane@sjsu.edu has been deleted from student records.
Student account with email rachel.zane@sjsu.edu has been deleted from login system.

```

Console Output

- search_student_by_email(): takes email as an input to search the student in *student_records.csv*.

login.csv	student_records.csv	new 2	new 1	faculty_data.csv	course_records.csv	test_course_records.csv	test_facu
984	Brianna, Henry, brandy39@example.org, "594 Catherine Way, West Katherineborough, PA 04678", 8466612661, data200, 1, F						
985	Jeffrey, Taylor, danielaustrin@example.com, "20625 Linda Dale Suite 389, Port Tiffanystad, NM 51195", 8918659393, data245, 1, F						
986	Jennifer, Jensen, dgilbert@example.com, "108 Jennifer Overpass Apt. 118, West Maryberg, TX 09583", 7756560489, data200, 1, F						
987	Brittany, Ibarra, fbennett@example.org, "423 Lisa Falls, West Cassandraview, NV 31087", 8782883500, data200, 1, F						
988	Justin, Roach, katherinemcdonald@example.org, "USS Gordon, FPO AF 42920", 9907336479, data220, 1, F						
989	Darren, Goodwin, kimberly21@example.com, "07347 Anderson Lane, New Sarah, SC 83235", 8080240461, data228, 1, F						
990	Shelia, Bowen, ronald10@example.com, "0395 Stephanie Tunnel Suite 658, Lake Jeffreyburgh, LA 75541", 9758831305, data202, 1, F						
991	Mark, Skinner, sschmidt@example.org, "051 Pearson Spur Apt. 656, North Edwardview, TX 66546", 8649397867, data255, 1, F						
992	John, Decker, theresavasquez@example.org, "84679 Stephens Center Suite 910, Smithmouth, IN 86618", 9925340453, data200, 1, F						
993	Alexis, Valencia, bergion@example.net, "4182 Johnston Camp Apt. 205, Port Tammyland, RI 45635", 6618405205, data245, 0, F						
994	Laura, Daniels, brian44@example.com, "42380 George Throughway Apt. 098, North Kim, CA 22320", 8687820873, data230, 0, F						
995	Billy, Tucker, corysantana@example.com, "68570 Hogan Fork Apt. 117, Lake Joseph, NJ 68289", 6191970140, data226, 0, F						
996	Nancy, Jones, dakotawelch@example.net, "6330 Faith Parks Suite 115, Lake Zacharyhaven, OK 70787", 6378499636, data220, 0, F						
997	Mark, Baker, drodgers@example.net, "180 Jessica Parkways, West Kayla, GA 43584", 6749721930, data228, 0, F						
998	Alex, Clay, graysean@example.org, "4699 Hampton Valleys Apt. 253, Williamshaven, FW 39382", 8375955826, data245, 0, F						
999	Michael, Smith, janetnewman@example.org, "5103 Ronnie Green Suite 455, Clineland, IL 88679", 7587695741, data255, 0, F						
1000	David, Hopkins, soniachandler@example.com, "16803 Spencer Estates, South Garytown, MP 06515", 7881817363, data228, 0, F						
1001	Steven, Barry, valerierodriguez@example.com, "35671 Pittman River, Lake Jeanville, HI 73261", 7129815440, data201, 0, F						
1002	Rachel, Zane, rachel.zane@sjsu.edu, San Jose, 9432600435, data200, 92, A+						
1003							

Records of 'soniachandler@example.com' in student_records.csv

```

login_system.py student_system.py X main.py course_system.py grade_system.py
student_system.py > Student > load_grades
8 class Student:
172 def search_student_by_email(self, email):
173     start_time = time.time()
174     self.view_student_records(email)
175     end_time = time.time()
176     elapsed_time = end_time - start_time
177     print(f"Time taken to search for the student record: {elapsed_time:.4f} seconds")
178 def add_course(self, email):
179     valid_course_ids = []
180     try:
181         with open(self.course_file, mode = 'r') as file:
182             reader = csv.reader(file)
183             for row in reader:
184                 valid_course_ids.append(row[1].lower().replace(" ", ""))
185     except FileNotFoundError:
186         print("Error: The course file was not found.")
187     return
188     first_name = last_name = address = phone_number = course_id = None
189     marks = 0
190     grade = 'F'
191     with open(self.student_records, mode = 'r') as file:
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
```

```

login_system.py student_system.py X main.py course_system.py grad
student_system.py > Student > load_grades
8 class Student:
178 def add_course(self, email):
179     valid_course_ids = []
180     try:
181         with open(self.course_file, mode = 'r') as file:
182             reader = csv.reader(file)
183             for row in reader:
184                 valid_course_ids.append(row[1].lower().replace(" ", ""))
185     except FileNotFoundError:
186         print("Error: The course file was not found.")
187     return
188     first_name = last_name = address = phone_number = course_id = None
189     marks = 0
190     grade = 'F'
191     with open(self.student_records, mode = 'r') as file:
192         reader = csv.reader(file)
193         rows = list(reader)
194         student_found = False
195         for row in rows:
196             if row[2] == email:
197                 first_name = row[0]

```

6. Delete course
7. View Course List
8. View Grade Lookup
9. View professor by course
10. Change Password
11. Return to main menu
Please choose an option (1/2/3/4/5/6/7/8/9/10/11): 5
Enter the email to add course: rachel.zane@sjsu.edu
Enter the course ID to enroll (eg, data200, data201): data202
Enter your marks for the course (between 0 and 100): 100
Course data202 added for Rachel Zane. Marks: 100, Grade: A+.
Do you want to add another course? (y/n): y
Enter the course ID to enroll (eg, data200, data201): data230
Enter your marks for the course (between 0 and 100): 87
Course data230 added for Rachel Zane. Marks: 87, Grade: A+.
Do you want to add another course? (y/n): n

Console Output

- drop_course(): user can delete course for a student in *student_records.csv*

```

login.csv student_records.csv X new 2 new 1 faculty_data.csv course_records.csv test_course_records.csv test_fa
984 Brianna, Henry, brandy39@example.org, "594 Catherine Way, West Katherineborough, PA 04678", 8466612661, data200, 1, F
985 Jeffrey, Taylor, danielauistin@example.com, "20625 Linda Dale Suite 389, Fort Tiffanystad, NM 51195", 8918659393, data245, 1, F
986 Jennifer, Jensen, dgilbert@example.com, "108 Jennifer Overpass Apt. 118, West Maryberg, TX 09583", 7756560489, data200, 1, F
987 Brittany, Ibarra, fbennett@example.org, "423 Lisa Falls, West Cassandraview, NV 31087", 8782883500, data200, 1, F
988 Justin, Roach, katherinemcdonald@example.org, "USS Gordon, FPO AP 42920", 9907336479, data220, 1, F
989 Darren, Goodwin, kimberly21@example.com, "07347 Anderson Lane, New Sarah, SC 83235", 8080240461, data228, 1, F
990 Shelia, Bowen, ronald10@example.com, "0395 Stephanie Tunnel Suite 658, Lake Jeffreyburgh, LA 75541", 9758831305, data202, 1, F
991 Mark, Skinner, sschmidt@example.org, "051 Pearson Spur Apt. 656, North Edwardview, TX 66546", 8649397867, data255, 1, F
992 John, Decker, theresavaquez@example.org, "84679 Stephens Center Suite 910, Smithmouth, IN 86618", 9925340453, data200, 1, F
993 Alexis, Valencia, bergjon@example.net, "4182 Johnston Camp Apt. 205, Port Tammyland, RI 45635", 6618405205, data245, 0, F
994 Laura, Daniels, brian44@example.com, "42380 George Throughway Apt. 098, North Kim, CA 22320", 8687820873, data230, 0, F
995 Billy, Tucker, corysantana@example.com, "68570 Hogan Fork Apt. 117, Lake Joseph, NJ 68289", 6191970140, data226, 0, F
996 Nancy, Jones, dakotaswelch@example.net, "6330 Faith Parks Suite 115, Lake Zacharyhaven, OK 70787", 6378499636, data220, 0, F
997 Mark, Baker, drodgers@example.net, "180 Jessica Parkways, West Kayla, GA 43584", 6749721930, data228, 0, F
998 Alex, Clay, graysean@example.org, "4699 Hampton Valleys Apt. 253, Williamshaven, PW 39382", 8375955826, data245, 0, F
999 Michael, Smith, janetnewman@example.org, "5103 Ronnie Green Suite 455, Clineland, IL 88679", 7587695741, data255, 0, F
1000 David, Hopkins, soniachandler@example.com, "16803 Spencer Estates, South Garytown, MP 06515", 7581817363, data228, 0, F
1001 Steven, Barry, valerierodriguez@example.com, "35671 Pittman River, Lake Jeanville, HI 73261", 7129815440, data201, 0, F
1002 Rachel, Zane, rachel.zane@sjsu.edu, San Jose, 9432600435, data202, 100, A+
1003 Rachel, Zane, rachel.zane@sjsu.edu, San Jose, 9432600435, data230, 87, A+
1004

```

Course 'data200' dropped for 'rachel.zane@sjsu.edu'

```

login_system.py student_system.py X main.py course_system.py grade_system.py
student_system.py > Student > load_grades
8 class Student:
225     def drop_course(self, email):
226         student_courses = []
227         first_name = last_name = address = phone_number = None
228         with open(self.student_records, mode = 'r') as file:
229             reader = csv.reader(file)
230             rows = list(reader)
231             for row in rows:
232                 if row[2] == email:
233                     if first_name is None:
234                         first_name = row[0]
235                         last_name = row[1]
236                         address = row[3]
237                         phone_number = row[4]
238                     student_courses.append(row)
239             if len(student_courses) == 0:
240                 print("No student found with this email in the records.")
241                 return
242             if len(student_courses) == 1:
243                 print("You should be enrolled in at least one course.\nCannot drop course.")
244             return

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

4. Search student by email
5. Add course
6. Delete course
7. View Course List
8. View Grade Lookup
9. View professor by course
10. Change Password
11. Return to main menu
Please choose an option (1/2/3/4/5/6/7/8/9/10/11): 6
Enter the email to drop course: rachel.zane@sjsu.edu
You are enrolled in the following courses:
1. data200 (Marks: 92, Grade: A+)
2. data202 (Marks: 100, Grade: A+)
3. data230 (Marks: 87, Grade: A+)
Enter the number of the course you want to drop: 1
Course data200 dropped successfully.

```

Console Output

- `view_courses()`: enables user to view the list of courses along with course name, course ID and credits stored in `course_records.csv`.

```

login_system.py student_system.py X main.py course_system.py
student_system.py > Student > load_grades
8 class Student:
268     def view_courses(self):
269         self.courses.display_courses()
270     def view_grade_lookup(self):

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

Please choose an option (1/2/3/4/5/6/7/8/9/10/11): 7

-----Course Details-----
Course Name          Course ID  Credits
-----
id>name              id          credits
Computational Programming for Data Analytics data200          9
Mathematics for Applied Data Science data202          9
Data Warehouse        data226          9
Database Technologies for Data Analytics data201          9
Mathematical Methods for Data Analytics data220          9
Big Data Technologies and Applications data228          9
Business Intelligence and Data Visualization data230          9
Machine Learning Technologies data245          9
Deep Learning Technologies data255          9
Advanced Big Data     data123          9

```

- `view_grade_lookup()`: allows user to view grade lookup stored in `grades.csv`.

The screenshot shows a code editor with three tabs: `login_system.py`, `student_system.py` (active), and `main.py`. The active tab shows the following code:

```
student_system.py > Student > load_grades
8 class Student:
270     def view_grade_lookup(self):
271         self.grades.view_grades()
```

Below the code editor is a terminal window with the following output:

```
-----Sub Student Menu-----
1. View your records
2. Update your information
3. Delete your account
4. Search student by email
5. Add course
6. Delete course
7. View Course List
8. View Grade Lookup
9. View professor by course
10. Change Password
11. Return to main menu
Please choose an option (1/2/3/4/5/6/7/8/9/10/11): 8
Grade Lookup
A+: Max Marks = 100, Min Marks = 80
A: Max Marks = 79, Min Marks = 70
B: Max Marks = 69, Min Marks = 60
C: Max Marks = 59, Min Marks = 50
D: Max Marks = 49, Min Marks = 40
E: Max Marks = 39, Min Marks = 35
F: Max Marks = 34, Min Marks = 0
```

- `view_professor_by_course()`: user can view a list of professors and their rank grouped by course ID.

```
login_system.py student_system.py X main.py
student_system.py > Student > load_grades
8 class Student:
272     def view_professor_by_course(self):
273         from faculty_system import Faculty
274         faculty = Faculty()
275         faculty.display_professors_by_course()

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Professors grouped by course ID:

Course ID: data200
Name: John Daniels, Rank: assistant professor
Name: Matthew Powell, Rank: assistant professor
Name: Lisa Swanson, Rank: assistant professor
Name: Michael Orr, Rank: assistant professor
Name: Christopher Cox, Rank: junior professor
Name: David Gibbs, Rank: head of department
Name: Sheldon Cooper, Rank: assistant professor

Course ID: data201
Name: Gary Lambert, Rank: assistant professor
Name: Daniel Ortiz, Rank: assistant professor
Name: Jonathan Oliver, Rank: assistant professor
Name: Felicia Mcdonald, Rank: assistant professor
Name: Brittany Anderson, Rank: junior professor
Name: Samuel Williams, Rank: head of department

Course ID: data202
Name: Tara Riley, Rank: assistant professor
Name: Blake Davis, Rank: assistant professor
Name: Sarah Thomas, Rank: assistant professor
Name: Steven Ellis, Rank: assistant professor
Name: Brianna Rodriguez, Rank: junior professor
Name: Crystal Gonzalez, Rank: head of department

Course ID: data220
Name: Katherine Taylor, Rank: assistant professor
Name: Courtney Rose, Rank: assistant professor
Name: Elizabeth Miles, Rank: assistant professor
Name: Victor Buchanan, Rank: assistant professor
Name: Michael Tran, Rank: junior professor
Name: Samantha Campbell, Rank: head of department
```

- **Course**
 - `display_courses()`: display the list of courses stored in `course_records.csv`


```

login_system.py student_system.py main.py course_system.py X grade_sy
course_system.py > Course > display_courses
1 import csv
2 class Course:
3     def __init__(self, course_file):
4         self.course_file = course_file
5     def display_courses(self):
6         try:
7             with open(self.course_file, mode = 'r') as file:
8                 reader = csv.reader(file)
9                 print("\n-----Course Details-----")
10                print(f"{'Course Name':<20} {'Course ID':<10} {'Credits':<7}")
11                print("-"*40)
12                for row in reader:
13                    course_name, course_id, credits = row
14                    print(f"{'course_name':<20} {'course_id':<20} {'credits':<7}")
15        except FileNotFoundError:
16            print(f"Error: The file '{self.course_file}' was not found.")
17        except Exception as e:
18            print(f"An error occurred: {e}")

```

- **Grade**

- load_grade_lookup(): reads the file *grades.csv*

```

login_system.py student_system.py main.py course_system.py grade_system.py X faculty_system.py
grade_system.py > Grade > view_grades
1 import csv
2 class Grade:
3     def __init__(self, grade_file = "C:/Users/aradh/Desktop/DATA 200 Python Programming/Lab1/grades.csv"):
4         self.grade_file = grade_file
5         self.grade_lookup = self.load_grade_lookup()
6     def load_grade_lookup(self):
7         grade_lookup = {}
8         try:
9             with open(self.grade_file, mode = 'r') as file:
10                reader = csv.reader(file)
11                next(reader)
12                for row in reader:
13                    if len(row) == 3:
14                        grade_lookup[row[0]] = {'max_marks': int(row[1]), 'min_marks': int(row[2])}
15        except FileNotFoundError:
16            print(f"Error: {self.grade_file} not found.")
17        except Exception as e:
18            print(f"An error occurred while loading grade lookup: {e}")
19        return grade_lookup

```

- view_grades(): user can view grade lookup

```

login_system.py student_system.py main.py course_system.py grade_system.py X faculty_sy
grade_system.py > Grade > view_grades
2 class Grade:
20     def view_grades(self):
21         if self.grade_lookup:
22             print("Grade Lookup")
23             for grade, info in self.grade_lookup.items():
24                 print(f"{'grade':<20} Max Marks = {info['max_marks']}, Min Marks = {info['min_marks']}")
25         else:
26             print("No grade information available.")

```

- **Faculty**

- display_faculty_info(): view a professor's information stored in *faculty_data.csv*

47	Carol,Wright,carol.wright@sjsu.edu,Robinsonton,4872410640,data230,assistant professor
48	Andrew,Mills,andrew.mills@sjsu.edu,Port Brettside,1714957578,data230,junior professor
49	Christy,Swanson,christy.swanson@sjsu.edu,West Stephanie,9444550354,data230,head of department
50	Deborah,Humphrey,deborah.humphrey@sjsu.edu,West Danielmouth,7111465926,data255,assistant professor
51	Jose,Rush,jose.rush@sjsu.edu,Youngmouth,5032036259,data255,assistant professor
52	Tiffany,Tran,tiffany.tran@sjsu.edu,Robinsonmouth,9276036313,data255,assistant professor
53	Laura,Long,laura.long@sjsu.edu,North Jenniferstad,6050705360,data255,assistant professor
54	Laura,Solis,laura.solis@sjsu.edu,South David,4872369563,data255,junior professor
55	Patrick,Soto,patrick.soto@sjsu.edu,East Elijah,4286177622,data255,head of department
56	Aradhya,Aggarwal,aradhya.aggarwal@sjsu.edu,East Elijah,4286177622,data255,head of department
57	Sheldon,Cooper,sheldon.cooper@sjsu.edu,San Jose,9432600578,data200,assistant professor
58	

Records for professor 'sheldon.cooper@sjsu.edu' in faculty_data.csv

```

login_system.py student_system.py main.py course_system.py grade_system.py faculty_
faculty_system.py > Faculty > sort_student_records
6 class Faculty:
19 def display_faculty_info(self, email):
20     found = False
21     try:
22         with open(self.faculty_file, mode = 'r') as file:
23             reader = csv.reader(file)
24             next(reader)
25             for row in reader:
26                 if len(row) == 7 and row[2].lower() == email.lower():
27                     first_name, last_name, email, address, phone_number, course_id, rank = row
28                     found = True
29                     print("Professor Information:")
30                     print(f"Name: {first_name} {last_name}")
31                     print(f"Email: {email}")
32                     print(f"Address: {address}")
33                     print(f"Phone Number: {phone_number}")
34                     print(f"Course ID: {course_id}")
35                     print(f"Rank: {rank}")
36                     break
37             if not found:
38                 print(f"No professor with the email: {email}")
39         except FileNotFoundError:
40             print(f"Error: The faculty file {self.faculty_file} was not found.")

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

12. Delete Course
13. Modify Grade Lookup
14. Delete your account
15. Change password
16. Return to main menu
Please choose an option (1/2/3/4/5/6/7/8/9/10/11/12/13/14/15/16): 1
Enter your email to view your records: sheldon.cooper@sjsu.edu
Professor Information:
Name: Sheldon Cooper
Email: sheldon.cooper@sjsu.edu
Address: San Jose
Phone Number: 9432600578
Course ID: data200
Rank: assistant professor

```

Console Output

- create_professor(): Once the user decides to sign-up as a new professor the function create_professor() creates a new record for the professor in faculty_data.csv

47	Carol,Wright,carol.wright@sjsu.edu,Robinsonton,4872410640,data230,assistant professor
48	Andrew,Mills,andrew.mills@sjsu.edu,Port Brettside,1714957578,data230,junior professor
49	Christy,Swanson,christy.swanson@sjsu.edu,West Stephanie,9444550354,data230,head of department
50	Deborah,Humphrey,deborah.humphrey@sjsu.edu,West Danielmouth,7111465926,data255,assistant professor
51	Jose,Rush,jose.rush@sjsu.edu,Youngmouth,5032036259,data255,assistant professor
52	Tiffany,Tran,tiffany.tran@sjsu.edu,Robinsonmouth,9276036313,data255,assistant professor
53	Laura,Long,laura.long@sjsu.edu,North Jenniferstad,6050705360,data255,assistant professor
54	Laura,Solis,laura.solis@sjsu.edu,South David,4872369563,data255,junior professor
55	Patrick,Soto,patrick.soto@sjsu.edu,East Elijah,4286177622,data255,head of department
56	Aradhya,Aggarwal,aradhya.aggarwal@sjsu.edu,East Elijah,4286177622,data255,head of department
57	Sheldon,Cooper,sheldon.cooper@sjsu.edu,San Jose,9432600578,data200,assistant professor
58	

Records for professor 'sheldon.cooper@sjsu.edu' created in faculty_data.csv

The screenshot shows a code editor with several tabs: login_system.py, student_system.py, main.py, course_system.py, grade_system.py, and faculty_system.py. The faculty_system.py tab is active, showing the Faculty class. The code defines a create_professor method that reads a CSV file to validate course IDs, prompts for user input (email, first name, last name, address, phone number), and checks if the email already exists. The terminal output shows the successful creation of an account for Sheldon Cooper, including prompts for email, name, address, phone number, and rank, followed by a confirmation message.

```

6 class Faculty:
43     def create_professor(self):
44         valid_course_ids = []
45         try:
46             with open(self.course_file, mode = 'r') as file:
47                 reader = csv.reader(file)
48                 next(reader)
49                 for row in reader:
50                     valid_course_ids.append(row[1].lower().replace(" ", ""))
51         except FileNotFoundError:
52             print(f"Error: The faculty file {self.course_file} was not found.")
53             return
54         except Exception as e:
55             print(f"An error occurred while reading the course file: {e}")
56             return
57         email = input("Enter your email: ")
58         if self.is_email_exists(email):
59             print(f"A professor with email {email} already exists.")
60             return
61         first_name = input("Enter your first name: ")
62         last_name = input("Enter your last name: ")
63         address = input("Enter your address: ")
64         phone_number = input("Enter your phone number: ")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Account created successfully for sheldon.cooper@sjsu.edu as a professor.
Enter your email: sheldon.cooper@sjsu.edu
Enter your first name: Sheldon
Enter your last name: Cooper
Enter your address: San Jose
Enter your phone number: 9432600578
Enter the course ID you will be teaching (eg. data200, data201): data200

Please choose your rank from the following options:
1. Junior Professor
2. Assistant Professor
3. Head of Department
Enter the number corresponding to your rank: 2
Account created successfully for Sheldon Cooper.

```

Console Output

- is_email_exists(): checks if the entered email is present in *faculty_data.csv*

The screenshot shows the code editor with the faculty_system.py tab active. The code defines the is_email_exists method, which reads the faculty_data.csv file to check if the entered email is already present. The method returns True if the email is found and False otherwise. It also includes error handling for file not found and other exceptions.

```

89     def is_email_exists(self, email):
90         try:
91             with open(self.faculty_file, mode = 'r') as file:
92                 reader = csv.reader(file)
93                 next(reader)
94                 for row in reader:
95                     if len(row) == 7 and row[2].lower() == email.lower():
96                         return True
97             return False
98         except FileNotFoundError:
99             print(f"Error: The faculty file {self.faculty_file} was not found.")
100             return False
101         except Exception as e:
102             print(f"An error while checking the email: {e}")
103             return False

```

- display_professors_by_course(): display the list of professors and their rank grouped by course ID

```

login_system.py student_system.py main.py course_system.py grade_system.py faculty_system.py
6 class Faculty:
104 def display_professors_by_course(self):
105     professors_by_course = {}
106     try:
107         with open(self.faculty_file, mode = 'r') as file:
108             reader = csv.reader(file)
109             next(reader)
110             for row in reader:
111                 first_name = row[0]
112                 last_name = row[1]
113                 email = row[2]
114                 course_id = row[5]
115                 rank = row[6]
116                 if course_id not in professors_by_course:
117                     professors_by_course[course_id] = []
118                     professors_by_course[course_id].append((first_name+" "+last_name, rank))
119     except FileNotFoundError:
120         print(f"Error: The file {self.faculty_file} was not found.")
121         return
122     except Exception as e:
123         print(f"An error occurred while reading the faculty records: {e}")
124         return
125     if not professors_by_course:

```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** PORTS

Professors grouped by course ID:

Course ID: data200
 Name: John Daniels, Rank: assistant professor
 Name: Matthew Powell, Rank: assistant professor
 Name: Lisa Swanson, Rank: assistant professor
 Name: Michael Orr, Rank: assistant professor
 Name: Christopher Cox, Rank: junior professor
 Name: David Gibbs, Rank: head of department
 Name: Sheldon Cooper, Rank: assistant professor

Course ID: data201
 Name: Gary Lambert, Rank: assistant professor
 Name: Daniel Ortiz, Rank: assistant professor
 Name: Jonathan Oliver, Rank: assistant professor

- modify_professor_info(): The “Check My Grade” application assumes that once a user has registered their first name, last name, email cannot be modified. However, details like phone number and address can change and thus allow the user to modify these details.

```

faculty_data.csv login.csv student_records.csv new 2 new 1 course_records.csv test_cou
47 Carol,Wright,carol.wright@sjsu.edu,Robinsonton,4872410640,data230,assistant professor
48 Andrew,Mills,andrew.mills@sjsu.edu,Port Brettside,1714957578,data230,junior professor
49 Christy,Swanson,christy.swanson@sjsu.edu,West Stephanie,9444550354,data230,head of department
50 Deborah,Humphrey,deborah.humphrey@sjsu.edu,West Danielmouth,7111465926,data255,assistant professor
51 Jose,Rush,jose.rush@sjsu.edu,Youngmouth,5032036259,data255,assistant professor
52 Tiffany,Tran,tiffany.tran@sjsu.edu,Robinsonmouth,9276036313,data255,assistant professor
53 Laura,Long,laura.long@sjsu.edu,North Jenniferstad,6050705360,data255,assistant professor
54 Laura,Solis,laura.solis@sjsu.edu,South David,4872369563,data255,junior professor
55 Patrick,Soto,patrick.soto@sjsu.edu,East Elijah,4286177622,data255,head of department
56 Aradhya,Aggarwal,aradhya.aggarwal@sjsu.edu,East Elijah,4286177622,data255,head of department
57 Sheldon,Cooper,sheldon.cooper@sjsu.edu,Livemrore,6502437812,data200,assistant professor
58

```

Details modified for professor 'sheldon.cooper@sjsu.edu'

```

login_system.py student_system.py main.py course_system.py grade_system.py faculty_system.py X
faculty_system.py > Faculty > sort_student_records
6 class Faculty:
133 def modify_professor_info(self, email):
134     found = False
135     rows = []
136     try:
137         with open(self.faculty_file, mode = 'r') as file:
138             reader = csv.reader(file)
139             rows = list(reader)
140     except FileNotFoundError:
141         print(f"Error: The file {self.faculty_file} was not found.")
142         return
143     except Exception as e:
144         print(f"An error occurred while reading the faculty records: {e}")
145         return
146     for i, row in enumerate(rows):
147         if row[2] == email:
148             found = True
149             print(f"Professor found: {row[0]} {row[1]}")
150             print(f"Current Address: {row[3]}")
151             print(f"Current Phone Number: {row[4]}")
152             new_address = input("Enter your new address: ")
153             new_phone = input("Enter your new phone number: ")
154             row[3] = new_address
155             row[4] = new_phone
156             print(f"Professor's information updated successfully.")
157         else:
158             continue
159     if not found:
160         print(f"Professor {email} not found.")
161     return rows

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
11. Add Course
12. Delete Course
13. Modify Grade Lookup
14. Delete your account
15. Change password
16. Return to main menu
Please choose an option (1/2/3/4/5/6/7/8/9/10/11/12/13/14/15/16): 3
Enter your email to modify records: sheldon.cooper@sjsu.edu
Professor found: Sheldon Cooper
Current Address: San Jose
Current Phone Number: 9432600578
Enter your new address: Livemore
Enter your new phone number: 6502437812
Professor's information updated successfully.

```

Console Output

- view_average_and_median_marks(): user can view average and median marks of each course scored by students. The output is sorted by average marks in decreasing order.

```

login_system.py student_system.py main.py course_system.py grade_system.py faculty_system.py X
faculty_system.py > Faculty > sort_student_records
6 class Faculty:
166 def view_average_and_median_marks(self):
167     course_marks = {}
168     try:
169         with open(self.student_records, mode = 'r') as file:
170             reader = csv.reader(file)
171             next(reader)
172             for row in reader:
173                 course_id = row[5]
174                 marks = int(row[6])
175                 if course_id not in course_marks:
176                     course_marks[course_id] = []
177                 course_marks[course_id].append(marks)
178     except FileNotFoundError:
179         print(f"Error: The file {self.student_records} was not found.")
180         return
181     except Exception as e:
182         print(f"An error occurred while reading the student records: {e}")
183         return
184     course_stats = []
185     for course_id, marks in course_marks.items():
186         try:
187             avg_marks = sum(marks)/len(marks)
188             median_marks = statistics.median(marks)
189             course_stats.append((course_id, avg_marks, median_marks))
190         except Exception as e:
191             print(f"Error processing course {course_id}: {e}")
192     start_time = time.time()
193     course_stats.sort(key = lambda x: x[1], reverse = True)
194     end_time = time.time()
195     elapsed_time = end_time - start_time
196     print(f"Time taken to sort the courses by average marks: {elapsed_time:.4f} seconds")
197     for course_id, avg_marks, median_marks in course_stats:
198         print(f"Course ID: {course_id}")
199         print(f"Average Marks: {avg_marks:.2f}")
200         print(f"Median Marks: {median_marks}\n")

```

```

Please choose an option (1/2/3/4/5/6/7/8/9/10/11/12/13/14/15/16): 4
Time taken to sort the courses by average marks: 0.0000 seconds
Course ID: data202
Average Marks: 53.42
Median Marks: 54.5

Course ID: data228
Average Marks: 53.09
Median Marks: 56.0

Course ID: data220
Average Marks: 53.01
Median Marks: 57

Course ID: data230
Average Marks: 50.90
Median Marks: 46.5

Course ID: data245
Average Marks: 50.42
Median Marks: 49.0

Course ID: data201
Average Marks: 49.57
Median Marks: 52.0

Course ID: data226
Average Marks: 48.49
Median Marks: 45

Course ID: data200
Average Marks: 45.83
Median Marks: 44

Course ID: data255
Average Marks: 43.73
Median Marks: 38

```

- `search_professor_by_email()`: user can view details of a particular professor based on the email entered.

	faculty_data.csv	login.csv	student_records.csv	new 2	new 1	course_records.csv	test_cou
35	Andrew, Perez, andrew.perez@sjsu.edu, 3805059333, data245, assistant professor						
36	Colin, Espinoza, colin.espinoza@sjsu.edu, Port Kevinberg, 4307455331, data245, junior professor						
37	Stephen, Smith, stephen.smith@sjsu.edu, New Anthony, 4660878874, data245, head of department						
38	Gary, Harmon, gary.harmon@sjsu.edu, Moorefurt, 7617185591, data228, assistant professor						
39	Joshua, Harris, joshua.harris@sjsu.edu, East Shane, 6331123628, data228, assistant professor						
40	Sarah, Russell, sarah.russell@sjsu.edu, Tapiastad, 0315915949, data228, assistant professor						
41	Daniel, Jackson, daniel.jackson@sjsu.edu, East Gabriel, 6559226722, data228, assistant professor						
42	Emily, Dalton, emily.dalton@sjsu.edu, West Lisa, 7590002851, data228, junior professor						
43	Mark, Miller, mark.miller@sjsu.edu, Kevinmouth, 9447349982, data228, head of department						
44	Leah, Phillips, leah.phillips@sjsu.edu, Michaelhaven, 5554400642, data230, assistant professor						
45	Frederick, Johnson, frederick.johnson@sjsu.edu, Brownhaven, 4259756656, data230, assistant professor						
46	Heather, Bailey, heather.bailey@sjsu.edu, Valeriestad, 5051123202, data230, assistant professor						
47	Carol, Wright, carol.wright@sjsu.edu, Robinsonton, 4872410640, data230, assistant professor						
48	Andrew, Mills, andrew.mills@sjsu.edu, Port Brettside, 1714957578, data230, junior professor						
49	Christy, Swanson, christy.swanson@sjsu.edu, West Stephanie, 9444550354, data230, head of department						
50	Deborah, Humphrey, deborah.humphrey@sjsu.edu, West Danielmouth, 7111465926, data255, assistant professor						
51	Jose, Rush, jose.rush@sjsu.edu, Youngmouth, 5032036259, data255, assistant professor						
52	Tiffany, Tran, tiffany.tran@sjsu.edu, Robinsonmouth, 9276036313, data255, assistant professor						
53	Laura, Long, laura.long@sjsu.edu, North Jenniferstad, 6050705360, data255, assistant professor						
54	Laura, Solis, laura.solis@sjsu.edu, South David, 4872369563, data255, junior professor						
55	Patrick, Soto, patrick.soto@sjsu.edu, East Elijah, 4286177622, data255, head of department						
56	Sheldon, Cooper, sheldon.cooper@sjsu.edu, Livemrore, 6502437812, data200, assistant professor						
57							

Record of professor 'tiffany.tran@sjsu.edu' in faculty_data.csv

```

login_system.py student_system.py main.py course_system.py grade_system.py faculty_system.py X
faculty_system.py > Faculty > sort_student_records
6 class Faculty:
201     def search_professor_by_email(self, email):
202         start_time = time.time()
203         self.display_faculty_info(email)
204         end_time = time.time()
205         elapsed_time = end_time - start_time
206         print(f"Time taken to search for the professor record: {elapsed_time:.4f} seconds")
207     def modify_marks(self, email):

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

16. Return to main menu
Please choose an option (1/2/3/4/5/6/7/8/9/10/11/12/13/14/15/16): 7
Enter the email to search for professor: tiffany.tran@sjsu.edu
Professor Information:
Name: Tiffany Tran
Email: tiffany.tran@sjsu.edu
Address: Robinsonmouth
Phone Number: 9276036313
Course ID: data255
Rank: assistant professor
Time taken to search for the professor record: 0.0029 seconds

```

Console Output

- modify_marks(): user can modify marks of a student for a particular course ID. Once the marks are modified a new grade is assigned based on the new marks.

```

faculty_data.csv login.csv student_records.csv new 2 new 1 course_records.csv test_course_records.csv test_fac
984 Brianna, Henry, brandy39@example.org, "594 Catherine Way, West Katherineborough, PA 04678", 8466612661, data200, 1, F
985 Jeffrey, Taylor, danielaustrin@example.com, "20625 Linda Dale Suite 389, Fort Tiffanystad, NM 51195", 8918659393, data245, 1, F
986 Jennifer, Jensen, dgilbert@example.com, "108 Jennifer Overpass Apt. 118, West Maryberg, TX 09583", 7756560489, data200, 1, F
987 Brittany, Ibarra, fbennett@example.org, "423 Lisa Falls, West Caseandraview, NV 31087", 8792883500, data200, 1, F
988 Justin, Roach, katherinemcdonald@example.org, "USS Gordon, FPO AF 42920", 9907336479, data220, 1, F
989 Darren, Goodwin, kimberly21@example.com, "07347 Anderson Lane, New Sarah, SC 83235", 8080240461, data228, 1, F
990 Shelia, Bowen, ronald10@example.com, "0395 Stephanie Tunnel Suite 658, Lake Jeffreyburgh, LA 75541", 9758831305, data202, 1, F
991 Mark, Skinner, sschmidt@example.org, "051 Pearson Spur Apt. 656, North Edwardview, TX 66546", 8649397867, data255, 1, F
992 John, Decker, theresavasquez@example.org, "84679 Stephens Center Suite 910, Smithmouth, IN 86618", 9925340453, data200, 1, F
993 Alexis, Valencia, bergjon@example.net, "4182 Johnston Camp Apt. 205, Port Tammyland, RI 45635", 6618405205, data245, 0, F
994 Laura, Daniels, brian44@example.com, "42380 George Throughway Apt. 098, North Kim, CA 22320", 8687820873, data230, 0, F
995 Billy, Tucker, corysantana@example.com, "68570 Hogan Fork Apt. 117, Lake Joseph, NJ 68289", 6191970140, data226, 0, F
996 Nancy, Jones, dakotawelch@example.net, "6330 Faith Parks Suite 115, Lake Zacharyhaven, OK 70787", 6378499636, data220, 0, F
997 Mark, Baker, drodgers@example.net, "180 Jessica Parkways, West Kayla, GA 43584", 6749721930, data228, 0, F
998 Alex, Clay, graysean@example.org, "4699 Hampton Valleys Apt. 253, Williamshaven, FW 39382", 8375955826, data245, 0, F
999 Michael, Smith, janetnewman@example.org, "5103 Ronnie Green Suite 455, Clineland, IL 88679", 7587695741, data255, 0, F
1000 David, Hopkins, soniachandler@example.com, "16803 Spencer Estates, South Garytown, MP 06515", 7881817363, data228, 0, F
1001 Steven, Barry, valerierodriguez@example.com, "35671 Pittman River, Lake Jeanville, HI 73261", 7129815440, data201, 0, F
1002 Rachel, Zane, rachel.zane@sjsu.edu, San Jose, 9432600435, data202, 100, A+
1003 Rachel, Zane, rachel.zane@sjsu.edu, San Jose, 9432600435, data230, 45, D
1004

```

Records for student 'rachel.zane@sjsu.edu' modified in student_records.csv

```

login_system.py student_system.py main.py course_system.py grade_system.py faculty_system.py X test.py
faculty_system.py > Faculty > sort_student_records
6 class Faculty:
207     def modify_marks(self, email):
208         students_in_course = []
209         try:
210             with open(self.student_records, mode='r') as file:
211                 reader = csv.reader(file)
212                 next(reader)
213                 for row in reader:
214                     if row[2].lower() == email.lower():
215                         students_in_course.append(row)
216         except FileNotFoundError:
217             print(f"Error: The student records file was not found.")
218             return
219         if not students_in_course:
220             print(f"No student found with email {email}.")
221             return
222         print(f"\nStudent found: {students_in_course[0][0]} {students_in_course[0][1]} (Email: {students_in_course[0][2]})")
223         print(f"Student is enrolled in the following courses:")
224         student_courses = set()
225         for row in students_in_course:
226             student_courses.add(row[5].lower())
227         for course_id in student_courses:
228             print(course_id)
229         course_id = input("Enter the course ID you want to modify marks for: ").lower().replace(" ", "")
230         if course_id.lower() not in student_courses:

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

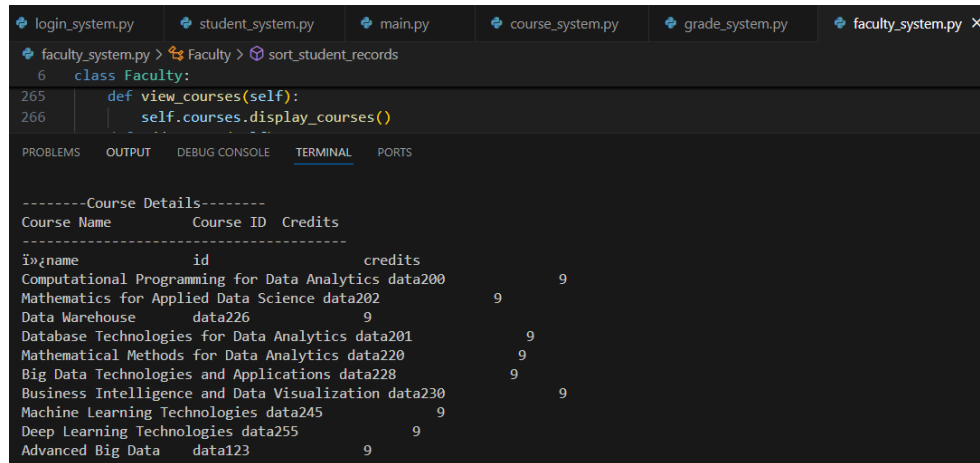
16. Return to main menu
Please choose an option (1/2/3/4/5/6/7/8/9/10/11/12/13/14/15/16): 8
Enter the email of the student for which you want to modify marks: rachel.zane@sjsu.edu
Enter the email of the student for which you want to modify marks: rachel.zane@sjsu.edu

Student found: Rachel Zane (Email: rachel.zane@sjsu.edu)
Student is enrolled in the following courses:
data230
data202
Enter the course ID you want to modify marks for: data230
Enter the new marks for rachel.zane@sjsu.edu in course data230: 45
Marks and grade updated successfully for rachel.zane@sjsu.edu in data230.

```

Console Output

- `view_courses()`: user can view list of available courses stored in `course_records.csv`.



```

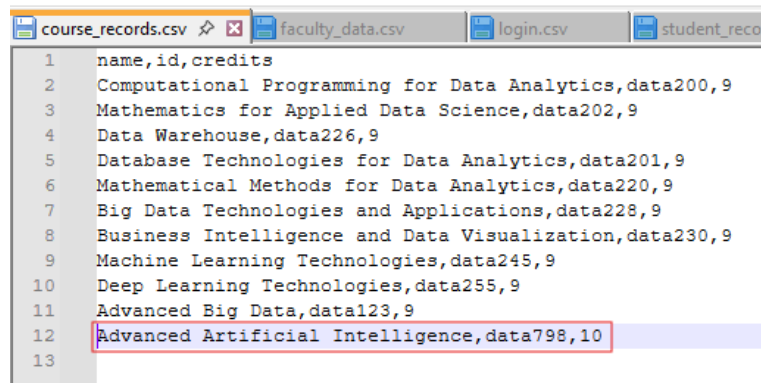
login_system.py student_system.py main.py course_system.py grade_system.py faculty_system.py
faculty_system.py > Faculty > sort_student_records
6 class Faculty:
265     def view_courses(self):
266         self.courses.display_courses()

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

-----Course Details-----
Course Name      Course ID  Credits
-----
name            id          credits
Computational Programming for Data Analytics data200      9
Mathematics for Applied Data Science data202      9
Data Warehouse   data226      9
Database Technologies for Data Analytics data201      9
Mathematical Methods for Data Analytics data220      9
Big Data Technologies and Applications data228      9
Business Intelligence and Data Visualization data230      9
Machine Learning Technologies data245      9
Deep Learning Technologies data255      9
Advanced Big Data   data123      9

```

- `add_course()`: The “Check My Grade” application assumes that addition of a course can only be decided by the Head of Department. The function `add_course()` enables user to add a new row in `course_records.csv` which contains course name, course ID and credits



```

course_records.csv faculty_data.csv login.csv student_records.csv
1 name,id,credits
2 Computational Programming for Data Analytics,data200,9
3 Mathematics for Applied Data Science,data202,9
4 Data Warehouse,data226,9
5 Database Technologies for Data Analytics,data201,9
6 Mathematical Methods for Data Analytics,data220,9
7 Big Data Technologies and Applications,data228,9
8 Business Intelligence and Data Visualization,data230,9
9 Machine Learning Technologies,data245,9
10 Deep Learning Technologies,data255,9
11 Advanced Big Data,data123,9
12 Advanced Artificial Intelligence,data798,10
13

```

New course added in 'course_records.csv'


```

login_system.py student_system.py main.py course_system.py grade_system.py faculty_system.py X
faculty_system.py > Faculty > sort_student_records
6 class Faculty:
267     def add_course(self):
268         course_name = input("Enter the course name you want to add: ")
269         course_id = input("Enter the course ID: ")
270         course_id = course_id.lower().replace(" ", "")
271         credits = input("Enter the number of credits: ")
272         course_exists = False
273         try:
274             with open(self.course_file, mode = 'r') as file:
275                 reader = csv.reader(file)
276                 next(reader)
277                 for row in reader:
278                     if row[1].lower() == course_id:
279                         course_exists = True
280                         break
281         except FileNotFoundError:
282             print(f"The course records file {self.course_file} was not found.")
283             return
284         if course_exists:
285             print(f"Course with ID {course_id} already exists. Please enter a valid course ID.")
286             return
287         with open(self.course_file, mode = 'a', newline = '') as file:
288             writer = csv.writer(file)
289             writer.writerow([course_name, course_id, credits])
290             print(f"Course {course_name} with ID {course_id} and {credits} has been added successfully.")

```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** PORTS

```

Enter the course ID: data798
Please choose an option (1/2/3/4/5/6/7/8/9/10/11/12/13/14/15/16): 11
Enter your email to add course: patrick.soto@sjsu.edu
Enter the course name you want to add: Advanced Artificial Intelligence
Enter the course ID: data798
Enter your email to add course: patrick.soto@sjsu.edu
Enter the course name you want to add: Advanced Artificial Intelligence
Enter the course ID: data798
Enter the course name you want to add: Advanced Artificial Intelligence
Enter the course ID: data798
Enter the course ID: data798
Enter the number of credits: 10
Course Advanced Artificial Intelligence with ID data798 and 10 has been added successfully.

```

Console Output

- `get_professor_rank()`: The “Check My Grade” application assumes that there are some actions like adding a new course, deleting an existing course, or modifying the grade lookup can only be performed by “Head of Department”. Therefore, function `get_professor_rank()` returns the rank (assistant professor/junior professor/head of department) of a professor.

```

login_system.py student_system.py main.py course_system.py grade_system.py faculty_system.py X
faculty_system.py > Faculty > sort_student_records
6 class Faculty:
291     def get_professor_rank(self,email):
292         try:
293             found = False
294             with open(self.faculty_file, mode = 'r') as file:
295                 reader = csv.reader(file)
296                 next(reader)
297                 for row in reader:
298                     if row[2] == email:
299                         found = True
300                         return row[6].strip().lower().replace(" ", "")
301             if found == False:
302                 print(f"Professor {email} was not found.")
303                 return
304         except FileNotFoundError:
305             print(f"Error: The faculty file {self.faculty_file} was not found.")
306         return None

```

- `delete_course()` – The "Check My Grade" application operates under the assumption that the deletion of a course is only permitted to the Head of Department. This functionality allows users to delete a course from the `course_records.csv` file based on the entered course ID. Once a course is deleted, the application assumes that students can no longer enroll in that course. As a result, any records containing the deleted course ID are automatically removed from `student_records.csv`. To ensure data integrity, if a student is enrolled only in the deleted course, their login details are removed from `login.csv`. However, if a student is enrolled in multiple courses, only the row containing the deleted course ID is removed.

For example, if course ID 'data201' is deleted from *course_records.csv*, students can no longer enroll in 'data201', and all corresponding entries in *student_records.csv* are deleted. Additionally, 'rachel.zane@sjsu.edu' is enrolled only in 'data201', her records are removed from *login.csv*. On the other hand, 'sheldon.cooper@sjsu.edu' is enrolled in 'data201', her records are removed from *login.csv*. On the other hand, 'sheldon.cooper@sjsu.edu' is enrolled in 'data200' as well, his record remains in *login.csv* because he is still enrolled in at least one active course.

First name	Last name	email	address	phone number	course id	marks	grades
Sheldon	Cooper	sheldon.cooper@sjsu.edu	San Jose	9432600435	data200	96	A
Sheldon	Cooper	sheldon.cooper@sjsu.edu	San Jose	9432600435	data201	96	A
Rachel	Zane	rachel.zane@sjsu.edu	San Jose	6503450124	data201	62	D

984	Brianna, Henry, brandy39@example.org, "594 Catherine Way, West Katherineborough, PA 04678", 8466612661, data200, 1, F
985	Jeffrey, Taylor, daniel.austin@example.com, "20625 Linda Dale Suite 389, Port Tiffanystad, NM 51195", 8918659393, data245, 1, F
986	Jennifer, Jensen, dgilbert@example.com, "108 Jennifer Overpass Apt. 118, West Maryberg, TX 09583", 7756560489, data200, 1, F
987	Brittany, Ibarra, fbennett@example.org, "423 Lisa Falls, West Cassandraview, NV 31087", 8782883500, data200, 1, F
988	Justin, Roach, katherinemcdonald@example.org, "USS Gordon, FPO AF 42920", 9907336479, data220, 1, F
989	Darren, Goodwin, kimberly21@example.com, "07347 Anderson Lane, New Sarah, SC 83235", 8080240461, data228, 1, F
990	Shelia, Bowen, ronald10@example.com, "0395 Stephanie Tunnel Suite 658, Lake Jeffreyburgh, LA 75541", 9758831305, data202, 1, F
991	Mark, Skinner, ssschmidt@example.org, "051 Pearson Spur Apt. 656, North Edwardview, TX 66546", 8649397867, data255, 1, F
992	John, Decker, theressavasquez@example.org, "84679 Stephens Center Suite 910, Smithmouth, IN 86618", 9925340453, data200, 1, F
993	Alexis, Valencia, bergjon@example.net, "4182 Johnston Camp Apt. 205, Port Tammyland, RI 45635", 6618405205, data245, 0, F
994	Laura, Daniels, brian11@example.com, "42380 George Throughway Apt. 098, North Kim, CA 22320", 8687820873, data230, 0, F
995	Billy, Tucker, corysantana@example.com, "68570 Hogan Fork Apt. 117, Lake Joseph, NJ 68289", 6191970140, data226, 0, F
996	Nancy, Jones, dakotawelch@example.net, "6330 Faith Parks Suite 115, Lake Zacharyhaven, OK 70787", 6378499636, data220, 0, F
997	Mark, Baker, drodgers@example.net, "180 Jessica Parkways, West Kayla, GA 43584", 6749721930, data228, 0, F
998	Alex, Clay, graysean@example.org, "4699 Hampton Valleys Apt. 253, Williamshaven, FW 39382", 8375955826, data245, 0, F
999	Michael, Smith, janetnewman@example.org, "5103 Ronnie Green Suite 455, Clineland, IL 88679", 7587695741, data255, 0, F
1000	David, Hopkins, soniachandler@example.com, "16803 Spencer Estates, South Garytown, MP 06515", 7881817363, data228, 0, F
1001	Steven, Barry, valerierodriguez@example.com, "35671 Pittman River, Lake Jeanville, HI 73261", 7129815440, data201, 0, F
1002	Rachel, Zane, rachel.zane@sjsu.edu, San Jose, 9432600435, data202, 100, A+
1003	Rachel, Zane, rachel.zane@sjsu.edu, San Jose, 9432600435, data230, 45, D
1004	Rachel, Zane, rachel.zane@sjsu.edu, San Jose, 9432600435, data798, 45, D
1005	Mike, Ross, mike.ross@sjsu.edu, San Jose, 9432600435, data798, 45, D
1006	

Student 'rachel.zane@sjsu.edu' and 'mike.ross@sjsu.edu' enrolled in 'data798'

1032	danny.murphy@sjsu.edu, 123456, professor
1033	matthew.jones@sjsu.edu, 123456, professor
1034	mary.sullivan@sjsu.edu, 123456, professor
1035	andrew.perez@sjsu.edu, 123456, professor
1036	colin.espinoza@sjsu.edu, 123456, professor
1037	stephen.smith@sjsu.edu, 123456, professor
1038	gary.harmon@sjsu.edu, 123456, professor
1039	joshua.harris@sjsu.edu, 123456, professor
1040	sarah.russell@sjsu.edu, 123456, professor
1041	daniel.jackson@sjsu.edu, 123456, professor
1042	emily.dalton@sjsu.edu, 123456, professor
1043	mark.miller@sjsu.edu, 123456, professor
1044	leah.phillips@sjsu.edu, 123456, professor
1045	frederick.johnson@sjsu.edu, 123456, professor
1046	heather.bailey@sjsu.edu, 123456, professor
1047	carol.wright@sjsu.edu, 123456, professor
1048	andrew.mills@sjsu.edu, 123456, professor
1049	christy.swanson@sjsu.edu, 123456, professor
1050	deborah.humphrey@sjsu.edu, 123456, professor
1051	jose.rush@sjsu.edu, 123456, professor
1052	tiffany.tran@sjsu.edu, 123456, professor
1053	laura.long@sjsu.edu, 123456, professor
1054	laura.solis@sjsu.edu, 123456, professor
1055	patrick.soto@sjsu.edu, 123456, professor
1056	sheldon.cooper@sjsu.edu, jqhtrj, professor
1057	rachel.zane@sjsu.edu, jqhtrj, student
1058	mike.ross@sjsu.edu, jqhtrj, student
1059	

Student 'rachel.zane@sjsu.edu' and 'mike.ross@sjsu.edu' present in login.csv

1	name, id, credits
2	Computational Programming for Data Analytics, data200, 9
3	Mathematics for Applied Data Science, data202, 9
4	Data Warehouse, data226, 9
5	Database Technologies for Data Analytics, data201, 9
6	Mathematical Methods for Data Analytics, data220, 9
7	Big Data Technologies and Applications, data228, 9
8	Business Intelligence and Data Visualization, data230, 9
9	Machine Learning Technologies, data245, 9
10	Deep Learning Technologies, data255, 9
11	Advanced Big Data, data123, 9
12	

Course 'data798' deleted from course_records.csv

	course_records.csv	faculty_data.csv	login.csv	student_records.csv	new 2	new 1	test_course_records.csv	test_fac
984	Brianna, Henry, brandy39@example.org, "594 Catherine Way, West Katherineborough, PA 04678", 8466612661, data200, 1, F							
985	Jeffrey, Taylor, danielaustrin@example.com, "20625 Linda Dale Suite 389, Port Tiffanystad, NM 51195", 8918659393, data245, 1, F							
986	Jennifer, Jensen, dgilbert@example.com, "108 Jennifer Overpass Apt. 118, West Maryberg, TX 09583", 7756560489, data200, 1, F							
987	Brittany, Ibarra, fbennett@example.org, "423 Lisa Falls, West Cassandraview, NV 31087", 8782883500, data200, 1, F							
988	Justin, Roach, katherinemcdonald@example.org, "USS Gordon, FFO AP 42920", 9907336479, data220, 1, F							
989	Darren, Goodwin, kimberly21@example.com, "07347 Anderson Lane, New Sarah, SC 83235", 8080240461, data228, 1, F							
990	Shelia, Bowen, ronald10@example.com, "0395 Stephanie Tunnel Suite 658, Lake Jeffreyburgh, LA 75541", 9758831305, data202, 1, F							
991	Mark, Skinner, sschmidt@example.org, "051 Pearson Spur Apt. 656, North Edwardview, TX 66546", 8649397867, data255, 1, F							
992	John, Decker, theresavasquez@example.org, "84679 Stephens Center Suite 910, Smithmouth, IN 86618", 9925340453, data200, 1, F							
993	Alexis, Valencia, bergjon@example.net, "4182 Johnston Camp Apt. 205, Port Tammyland, RI 45635", 6618405205, data245, 0, F							
994	Laura, Daniels, brian44@example.com, "42380 George Throughway Apt. 098, North Kim, CA 22320", 8687820873, data230, 0, F							
995	Billy, Tucker, corysantana@example.com, "68570 Hogan Fork Apt. 117, Lake Joseph, NJ 68289", 6191970140, data226, 0, F							
996	Nancy, Jones, dakotawelch@example.net, "6330 Faith Parks Suite 115, Lake Zacharyhaven, OK 70787", 6378499636, data220, 0, F							
997	Mark, Baker, drodgers@example.net, "180 Jessica Parkways, West Kayla, GA 43584", 6749721930, data228, 0, F							
998	Alex, Clay, graysean@example.org, "4699 Hampton Valleys Apt. 253, Williamshaven, FW 39382", 8375955826, data245, 0, F							
999	Michael, Smith, janetnewman@example.org, "5103 Ronnie Green Suite 455, Clineland, IL 88679", 7587695741, data255, 0, F							
1000	David, Hopkins, soniachandler@example.com, "16803 Spencer Estates, South Garytown, MP 06515", 7881817363, data228, 0, F							
1001	Steven, Barry, valerierodriguez@example.com, "35671 Pittman River, Lake Jeanville, HI 73261", 7129815440, data201, 0, F							
1002	Rachel, Zane, rachel.zane@sjsu.edu, San Jose, 9432600435, data202, 100, A+							
1003	Rachel, Zane, rachel.zane@sjsu.edu, San Jose, 9432600435, data230, 45, D							
1004								

Student records for 'mike.ross@sjsu.edu' deleted from student_records.csv

	course_records.csv	faculty_data.csv	login.csv
1032	danny.murphy@sjsu.edu, 123456, professor		
1033	matthew.jones@sjsu.edu, 123456, professor		
1034	mary.sullivan@sjsu.edu, 123456, professor		
1035	andrew.perez@sjsu.edu, 123456, professor		
1036	colin.espinoza@sjsu.edu, 123456, professor		
1037	stephen.smith@sjsu.edu, 123456, professor		
1038	gary.harmon@sjsu.edu, 123456, professor		
1039	joshua.harris@sjsu.edu, 123456, professor		
1040	sarah.russell@sjsu.edu, 123456, professor		
1041	daniel.jackson@sjsu.edu, 123456, professor		
1042	emily.dalton@sjsu.edu, 123456, professor		
1043	mark.miller@sjsu.edu, 123456, professor		
1044	leah.phillips@sjsu.edu, 123456, professor		
1045	frederick.johnson@sjsu.edu, 123456, professor		
1046	heather.bailey@sjsu.edu, 123456, professor		
1047	carol.wright@sjsu.edu, 123456, professor		
1048	andrew.mills@sjsu.edu, 123456, professor		
1049	christy.swanson@sjsu.edu, 123456, professor		
1050	deborah.humphrey@sjsu.edu, 123456, professor		
1051	jose.rush@sjsu.edu, 123456, professor		
1052	tiffany.tran@sjsu.edu, 123456, professor		
1053	laura.long@sjsu.edu, 123456, professor		
1054	laura.solis@sjsu.edu, 123456, professor		
1055	patrick.soto@sjsu.edu, 123456, professor		
1056	sheldon.cooper@sjsu.edu, jqhtrj, professor		
1057	rachel.zane@sjsu.edu, jqhtrj, student		
1058			

Student 'mike.ross@sjsu.edu' deleted from login.csv as well

```

login_system.py × student_system.py main.py course_system.py grade_system.py faculty_system.py ×
faculty_system.py > Faculty > sort_student_records
6 class Faculty:
307     def delete_course(self, course_id):
308         course_id = course_id.lower().replace(" ", "")
309         course_exists = False
310         try:
311             with open(self.course_file, mode = 'r') as file:
312                 reader = csv.reader(file)
313                 next(reader)
314                 for row in reader:
315                     if row[1].lower().replace(" ", "") == course_id:
316                         course_exists = True
317                         break
318         except FileNotFoundError:
319             print(f"Error: The course file '{self.course_file}' was not found.")
320             return
321         if not course_exists:
322             print(f"Error: Course with ID '{course_id}' does not exist in the course records.")
323             return
324         try:
325             course_rows = []
326             with open(self.course_file, mode = 'r') as file:
327                 reader = csv.reader(file)
328                 course_rows = list(reader)
329             course_rows = [row for row in course_rows if row[1] != course_id]
330             with open(self.course_file, mode = 'w', newline = '') as file:
331                 writer = csv.writer(file)
332                 writer.writerow(course_rows)
333         except Exception as e:
334             print(f"Error: {e}")
335             return
336         print(f"Course {course_id} has been deleted from course_records.csv")
337         return

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
11. Add Course
12. Delete Course
13. Modify Grade Lookup
14. Delete your account
15. Change password
16. Return to main menu
Please choose an option (1/2/3/4/5/6/7/8/9/10/11/12/13/14/15/16): 12
Enter your email to delete course: patrick.soto@sjsu.edu
Enter the course ID you want to delete: data798
Course data798 has been deleted from course_records.csv
Student account with email mike.ross@sjsu.edu has been deleted from student records.
Student account with email mike.ross@sjsu.edu has been deleted from login system.

```

Console Output

- `load_data()`: load data from *student_records.csv*

```

login_system.py × student_system.py main.py course_system.py grade_system.py faculty_system.py ×
faculty_system.py > Faculty > sort_student_records
6 class Faculty:
375     def load_data(self):
376         student = []
377         with open(self.student_records, mode='r', encoding='utf-8-sig') as file:
378             reader = csv.DictReader(file)
379             for row in reader:
380                 cleaned_row = {key.strip(): value for key, value in row.items()}
381                 student.append({
382                     'first_name': cleaned_row['first_name'],
383                     'last_name': cleaned_row['last_name'],
384                     'email': cleaned_row['email'],
385                     'address': cleaned_row['address'],
386                     'phone_number': cleaned_row['phone_number'],
387                     'course_id': cleaned_row['course_id'],
388                     'marks': int(cleaned_row['marks'])
389                 })
390         return student

```

- `get_grade_lookup()`: The “Check My Grade” application assumes that only Head of Department can modify grade lookup stored in *grades.csv*. The function `get_grade_lookup()` allows user to input new grade lookup.

```

login_system.py × student_system.py main.py course_system.py grade_system.py faculty_system.py ×
faculty_system.py > Faculty > get_grade_lookup
6 class Faculty:
391 def get_grade_lookup(self):
392     grades = ['A+', 'A', 'B', 'C', 'D', 'E', 'F']
393     grade_lookup = []
394     print("Please enter the grade ranges for each grade:")
395     while True:
396         try:
397             minimum_A_plus = int(input("Enter the minimum marks for grade A+: "))
398             if minimum_A_plus < 0 or minimum_A_plus > 100:
399                 print("Error: The minimum for grade A+ must be between 0 and 100.")
400                 continue
401             grade_lookup.append({'grade': 'A+', 'minimum': minimum_A_plus, 'maximum': 100})
402             break
403         except ValueError:
404             print("Error: Please enter a valid integer for the minimum marks of A+.")
405     previous_minimum = grade_lookup[0]['minimum']
406     for i, grade in enumerate(grades[1:-1]):
407         while True:
408             try:
409                 minimum = int(input(f"Enter the minimum marks for grade {grade}: "))
410                 maximum = previous_minimum - 1
411                 print(f"Maximum marks for grade {grade}: {maximum}")
412                 if maximum < minimum:
413                     print(f"Error: Maximum marks for {grade} must be greater than or equal to minimum marks.")
414                     continue
415                 grade_lookup.append({'grade': grade, 'minimum': minimum, 'maximum': maximum})
416                 previous_minimum = minimum
417                 break
418             except ValueError:
419                 print("Error: Please enter valid integer values for minimum and maximum.")
420     maximum_F = previous_minimum - 1
421     if maximum_F <= 0:
422         print(f"Error: Invalid maximum {maximum_F} for grade F. Maximum must be a positive number.")
423     else:
424         grade_lookup.append({'grade': 'F', 'minimum': 0, 'maximum': maximum_F})
425     return grade_lookup

```

```

login_system.py × student_system.py main.py course_system.py grade_system.py faculty_system.py ×
faculty_system.py > Faculty > get_grade_lookup
6 class Faculty:
391 def get_grade_lookup(self):
392     grades = ['A+', 'A', 'B', 'C', 'D', 'E', 'F']
393     grade_lookup = []
394     print("Please enter the grade ranges for each grade:")
395     while True:
396         try:
397             minimum_A_plus = int(input("Enter the minimum marks for grade A+: "))
398             if minimum_A_plus < 0 or minimum_A_plus > 100:
399                 print("Error: The minimum for grade A+ must be between 0 and 100.")
400                 continue
401             grade_lookup.append({'grade': 'A+', 'minimum': minimum_A_plus, 'maximum': 100})
402             break
403         except ValueError:
404             print("Error: Please enter a valid integer for the minimum marks of A+.")
405     previous_minimum = grade_lookup[0]['minimum']
406     for i, grade in enumerate(grades[1:-1]):
407         while True:
408             try:
409                 minimum = int(input(f"Enter the minimum marks for grade {grade}: "))
410                 maximum = previous_minimum - 1

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

Please choose an option (1/2/3/4/5/6/7/8/9/10/11/12/13/14/15/16): 13
Enter your email to modify grade lookup: patrick.soto@sjsu.edu
Please enter the grade ranges for each grade:
Enter the minimum marks for grade A+: 95
Enter the minimum marks for grade A: 90
Maximum marks for grade A: 94
Enter the minimum marks for grade B: 80
Maximum marks for grade B: 89
Enter the minimum marks for grade C: 70
Maximum marks for grade C: 79
Enter the minimum marks for grade D: 60
Maximum marks for grade D: 69
Enter the minimum marks for grade E: 50
Maximum marks for grade E: 59
Student records updated and saved to C:/Users/aradh/Desktop/DATA 200 Python Programming/Lab1/student_records.csv.
Modified grade lookup saved to 'C:/Users/aradh/Desktop/DATA 200 Python Programming/Lab1/grades.csv' in the correct order.

```

- `check_overlap()`: detects if there is any overlap between the interval of marks for two grades. This approach ensures that no two grades are assigned for the same marks.
For example, a user can enter range for Grade A 90 – 100 and range for Grade A+ 95 – 100. The range 95 – 100 has grades A as well as A+ allotted to it. This scenario is prevented by checking overlap in the ranges entered.

```
login_system.py × student_system.py main.py course_system.py grade_system.py faculty_system.py ×
faculty_system.py > Faculty > get_grade_lookup
6 class Faculty:
426     def check_overlap(self, grades):
427         grades.sort(key=lambda x: x['minimum'])
428         for i in range(1, len(grades)):
429             if grades[i]['minimum'] <= grades[i-1]['maximum']:
430                 print(f'Overlap detected between {grades[i-1]['grade']} and {grades[i]['grade']}")
431                 return False
432         return True
```

- `assign_grade()`: The function `assign_grade()` allocates grades according to the grade lookup present in `grades.csv`.

```
login_system.py × student_system.py main.py course_system.py grade_system.py faculty_system.py ×
faculty_system.py > Faculty > get_grade_lookup
6 class Faculty:
433     def assign_grade(self, marks, grades):
434         for grade in grades:
435             if grade['minimum'] <= marks <= grade['maximum']:
436                 return grade['grade']
437         return 'F'
```

- `update_student_grades()`: This function updates the student grades based on the new grade lookup entered by the user.

```
login_system.py student_system.py main.py course_system.py grade_system.py faculty_system.py ×
faculty_system.py > Faculty > get_grade_lookup
6 class Faculty:
438     def update_student_grades(self, students, grades):
439         for student in students:
440             student['grade'] = self.assign_grade(student['marks'], grades)
441         return students
```

- `save_updated_data()`: Writes data back to `student_records.csv` once the grades are updated based on the new grade lookup.

grades.csv	course_records.csv	faculty_data.csv	login.csv	student_records.csv	new 2	new 1	test_course_records.csv
121				David,Proctor,xscott@example.com,"5351 Baird Spurs Apt. 482, West Douglasfurt, WY 47531",9880946759,data202,89,A+			
122				Jessica,Brice,bergmarie@example.net,"318 Davis Shore, Jamesshire, DC 20700",6389322775,data245,88,A+			
123				Albert,Smith,courtney74@example.net,"6657 Campbell Ranch, South Emilyberg, DE 01239",9546181015,data200,88,A+			
124				Lynn,Baker,kknapp@example.com,"8210 Beverly Dale, Garyview, CO 99885",7385378696,data200,88,A+			
125				Sandra,Kelly,nfreeman@example.net,"30344 Holmes Village, Watkinschester, PA 55120",8013481521,data200,88,A+			
126				Michael,Bradshaw,randolphpaula@example.net,"9272 Young Divide Suite 852, Williamsmouth, MH 07295",8695131085,data200,88,A+			
127				Allison,Watson,xcross@example.org,"904 Thompson Tunnel Apt. 077, New Rhonda, HI 92880",8184601690,data230,88,A+			
128				Kyle,Macias,roberthughes@example.net,"07938 Massey Place, Lindsayhaven, FL 63879",9119910149,data245,88,A+			
129				David,Johnson,ybryant@example.com,"416 Emily Lodge, Jeffreyside, VI 83972",9097120230,data228,88,A+			
130				Anthony,Le,bishopapril@example.org,"93085 Gilbert Knoll Apt. 947, Owensaside, SD 75903",8136927907,data245,87,A+			
131				Steve,Scott,blee@example.com,"0559 Morrison Harbors, Suttonport, NM 09655",9133882471,data255,87,A+			
132				Nicole,Love,brendahurst@example.org,"853 Wise Keys, Lake Amyshire, NV 29846",8426708519,data202,87,A+			
133				Maxwell,Avila,cjohnson@example.com,"PSC 1299, Box 6214, APO AP 69486",5375213145,data226,87,A+			
134				Scott,Jones,goodmanjami@example.net,"689 Brown Shoel Suite 440, Johnsonsland, UT 31090",6389960779,data228,87,A+			
135				Daniel,Perez,balleynest@example.com,"35014 Moore Hollow Apt. 650, Schneiderberg, NH 61827",7691991674,data230,86,A+			
136				Christopher,Young,hmorton@example.net,"5777 Chambers Harbors, East Annmouth, MS 50386",6595790086,data202,86,A+			
137				Katherine,Munoz,kimberly85@example.com,"601 Matthew Brook Suite 919, West Katrinamouth, CT 43470",7393364650,data226,86,A+			
138				William,Simons,lmiles@example.com,"80490 Arnold Avenue, Markchester, FW 37368",9153087041,data228,86,A+			
139				Regina,Davis,rachevelasquez@example.com,"1724 Sullivan Springs Suite 123, North Stephanie, ND 44162",6137379142,data245,86,A+			
140				Ryan,Espinosa,russellcochran@example.net,"00777 Taylor Land, North Zachary, WI 70344",7094135735,data226,86,A+			
141				Kimberly,Brown,taral1@example.net,"2585 Brittany Rest, Robertland, MD 57608",6239186610,data202,86,A+			
142				Brian,Brown,ynash@example.org,"269 Amber Hills Suite 439, West Vickie, SC 22228",8260248699,data201,86,A+			
143				Jennifer,Bishop,chelsea03@example.org,"5872 Eric Crossing Apt. 080, New James, IL 03999",7596646936,data202,85,A+			
144				Dawn,Garcia,imcgrath@example.org,"1930 Krista Parkways, West Josephbury, CO 97341",8457325089,data220,85,A+			
145				James,Lopez,joseph06@example.com,"872 Robinson Loaf Apt. 710, West Michaelfurt, KY 85632",7163489888,data245,85,A+			
146				Henry,Berry,martinezlance@example.net,"297 Katherine Fork Suite 927, Harperburgh, FW 97946",8234712356,data230,85,A+			
147				Joshua,Franklin,melissa49@example.org,"20392 Jordan Bridge, Lake Larry, GU 47200",7034430055,data230,85,A+			
148				Gregory,Garrett,nburke@example.net,"23213 Chad Parkways Suite 184, Johnbury, OK 45947",8705230322,data230,85,A+			
149				Patricia,Rodriguez,andrew42@example.org,"503 Cohen Turnpike, Mooreville, MH 95009",6525485189,data200,84,A+			
150				Jamie,Smith,ashleyclark@example.net,"7930 Hayden Oval, Cathyfurf, NJ 30825",6742030347,data228,84,A+			
151				Jason,Fernandez,davilalarry@example.net,"45725 Ferrell Roads, West Thomas, FR 65657",8290878966,data202,84,A+			
152				Jaqueline,Clayton,fsanchez@example.net,"190 Cox Tunnel, Tinafort, ID 83841",6739453588,data230,84,A+			
153				Todd,Mitchell,howardkevin@example.com,"00167 Charles Courts Suite 496, Jamesview, DE 64426",7282244652,data200,84,A+			
154				Catherine,Chen,jesuslee@example.com,"738 Brian Ferry, Nicholastown, DE 17147",6007660496,data220,84,A+			
155				Joshua,Hill,johnsonmegan@example.com,"6282 Cross Turnpike Suite 615, Meganfort, MD 87419",9714021226,data230,84,A+			
156				Mariessa,Hunter,kluna@example.org,"6341 Leslie Canyon Apt. 162, Sheppardborough, WV 77128",9012488577,data220,84,A+			
157				Jerry,Stevens,mcintoshpamela@example.net,"USNV Henry, FPO AE 15903",6705543838,data202,84,A+			
158				Matthew,Brown,melodyfisher@example.net,"502 Morales Shores Apt. 135, Michaeliland, NH 70251",7816325864,data220,84,A+			
159				Nicole,Walls,montgomerylinda@example.com,"38915 Patricia Lookas, Fort Williamhaven, FW 66355",9436799285,data230,84,A+			
160				Jerry,Elliott,moodyspaul@example.org,"20378 Sarah Dam Apt. 883, New Andrewview, NV 40070",9272142588,data201,84,A+			
161				Patricia,Deleon,nkidd@example.org,"PSC 3365, Box 6037, APO AP 18774",6270168759,data226,84,A+			
162				Cynthia,Bass,williams@example.com,"00447 Danny Overpass, Lunabury, WV 33833",9558316066,data202,84,A+			
163				Brenda,Hunter,sullivanhaley@example.net,"99531 Tamara Course, Derrickside, NV 44464",9822767017,data226,84,A+			
164				Daniel,Moore,amber34@example.com,"7915 Christian Island, Hansonstad, PR 26127",8179916965,data220,83,A+			
165				Benjamin,Oconnor,austin75@example.net,"8422 Turner View Apt. 882, Port Stacy, AK 99374",7195207475,data220,83,A+			
166				Kevin,Estrada,candice22@example.org,"953 Page Lodge Suite 218, New Tammy, FW 55898",8296519572,data226,83,A+			
167				Adam,Phillips,ecarter@example.net,"Unit 8347 Box 4074, DPO AE 62072",9937070077,data220,83,A+			
168				Joseph,Benton,jennifer12@example.com,"1696 Blanchard Valleys, Williamsborough, TX 82406",6752287834,data228,83,A+			
169				Sandra,Harper,jeremygarrett@example.com,"4735 Justin Freeway, New Jay, VI 09859",6986781701,data202,83,A+			
170				James,Campos,kaitlyn85@example.net,"574 Larsen View, West Jeremyland, SC 03652",6011685870,data202,83,A+			
171				Walter,Wright,nicolelewis@example.org,"7873 Brett Flat, West Margaretchester, AL 90327",9411146917,data230,83,A+			
172				Samantha,Collins,renee12@example.net,"15320 Diana Expressway Apt. 605, Jacksonchester, NC 91229",7190600777,data245,83,A+			
173				James,Murphy,brandonphillips@example.com,"444 Joshua Cape Apt. 091, Masonburgh, CA 02052",6329632198,data255,82,A+			

Original student_records.csv

121	David,Proctor,xacott@example.org,"5351 Baird Spurs Apt. 482, West Douglasfurt, WY 47531",9880946759,data202,89,B
122	Jessica,Price,bergmarie@example.net,"318 Davis Shore, Jamesshire, DC 20700",6389322775,data245,88,B
123	Albert,Smith,courtney74@example.net,"6657 Campbell Ranch, South Emilyberg, DE 01239",9546181015,data200,88,B
124	Lynn,Baker,knapp@example.com,"8210 Beverly Dale, Garyview, CO 99885",7385378696,data200,88,B
125	Sandra,Kelly,nfreeman@example.net,"30344 Holmes Village, Watkinschester, PA 55120",8013481521,data200,88,B
126	Michael,Bradshaw,randolphpaula@example.net,"9272 Young Divide Suite 852, Williamsmouth, MH 07295",8695131085,data200,88,B
127	Allison,Watson,rcross@example.org,"904 Thompson Tunnel Apt. 077, New Rhonda, HI 92880",8184601690,data230,88,B
128	Kyle,Macias,roberthughes@example.net,"07938 Massey Place, Lindsayhaven, FL 63879",9119910149,data245,88,B
129	David,Johnson,ybryant@example.com,"416 Emily Lodge, Jeffreyside, VI 83972",9097120230,data228,88,B
130	Anthony,Le,bishopapril@example.org,"93085 Gilbert Knoll Apt. 947, Owenside, SD 75903",8136927907,data245,87,B
131	Steve,Scott,blee@example.com,"0559 Morrison Harbors, Suttonport, NH 09655",9133882471,data255,87,B
132	Nicole,Love,brendashurt@example.org,"853 Wise Keys, Lake Amyshire, NY 29846",8426708519,data202,87,B
133	Maxwell,Avila,cjohnson@example.com,"PSC 1299, Box 6214, APO AP 69486",9373213148,data228,87,B
134	Scott,Jones,goodmanjamie@example.net,"695 Brown Shoal Suite 440, Johnsonstad, UT 31090",8389960779,data228,87,B
135	Daniel,Perez,baileyernest@example.com,"35014 Moore Hollow Apt. 650, Schneiderberg, NH 61827",7691991674,data230,86,B
136	Christopher,Young,hmorton@example.net,"5777 Chambers Harbors, East Annmouth, MS 50386",6595790086,data202,86,B
137	Katherine,Munoz,kimberly95@example.com,"601 Matthew Brook Suite 919, West Katrinamouth, CT 43470",7393364650,data226,86,B
138	William,Simpson,lmiles@example.com,"80490 Arnold Avenue, Markchester, FW 37369",9153087041,data228,86,B
139	Regina,Davis,rachelvelasquez@example.com,"1724 Sullivan Springs Suite 123, North Stephanie, ND 44162",6137379142,data245,86,B
140	Ryan,Espinosa,russellcoochran@example.net,"00777 Taylor Land, North Zachary, WI 70344",7094135735,data226,86,B
141	Kimberly,Brown,taral@example.net,"2585 Brittany Rest, Robertland, MD 57608",6239186610,data202,86,B
142	Brian,Brown,ynash@example.org,"269 Amber Hills Suite 439, West Vickie, SC 21228",8260248699,data201,86,B
143	Jennifer,Bishop,chelsea03@example.org,"5572 Eric Crossing Apt. 080, New James, IL 03999",7596646936,data202,85,B
144	Dawn,Garcia,mcgrath@example.org,"1930 Krista Parkways, West Josephbury, CO 97341",8457325089,data220,85,B
145	James,Lopez,joseph06@example.com,"872 Robinson Loaf Apt. 710, West Michaelfurt, KY 85632",7163489888,data245,85,B
146	Henry,Berry,martinezlance@example.net,"287 Katherine Fork Suite 927, Harperburgh, FW 97946",8234712356,data230,85,B
147	Joshua,Franklin,melissa49@example.org,"20982 Jordan Bridge, Lake Larry, GU 47200",7034430055,data230,85,B
148	Gregory,Garrett,nburke@example.net,"23213 Chad Parkways Suite 184, Johnbury, OK 45947",8705230322,data230,85,B
149	Patricia,Rodriguez,andrew42@example.org,"503 Cohen Turnpike, Mooreville, MH 95009",6525485189,data200,84,B
150	Jamie,Smith,ashleyclark@example.net,"7930 Hayden Oval, Cathyfurt, NJ 30825",6742030347,data228,84,B
151	Jason,Fernandez,davilalarry@example.net,"45725 Ferrell Roads, West Thomas, FR 65657",8290878966,data202,84,B
152	Jacqueline,Clayton,fsanchez@example.net,"190 Cox Tunnel, Tinafort, ID 83841",6739453588,data230,84,B
153	Todd,Mitchell,howardkevin@example.com,"00167 Charles Courts Suite 496, Jamesview, DE 64426",7282244652,data200,84,B
154	Catherine,Chen,jesulee@example.com,"7139 Brian Ferry, Nicholantown, DE 17147",6007660496,data220,84,B
155	Joshua,Hill,johnsonmegan@example.com,"6282 Cross Turnpike Suite 615, Neganfort, MD 87419",9714021226,data230,84,B
156	Marissa,Hunter,kluna@example.org,"6341 Leslie Canyon Apt. 162, Sheppardborough, WV 77128",9012488577,data220,84,B
157	Jerry,Stevens,mointoshpamela@example.net,"USNV Henry, FFO AE 15903",6705543838,data202,84,B
158	Matthew,Brown,melodyfisher@example.net,"502 Morales Shores Apt. 135, Michaellland, NH 70251",7816325864,data220,84,B
159	Nicole,Walls,montgomerylinda@example.com,"38915 Patricia Locks, Port Williamhaven, FL 66355",9436799288,data230,84,B
160	Jerry,Elliott,moodyspril@example.org,"08378 Sarah Dam Apt. 883, New Andrewview, NV 40070",9272142588,data201,84,B
161	Patricia,Deleon,nkidd@example.org,"PSC 3365, Box 6037, APO AP 18774",6270168759,data226,84,B
162	Cynthia,Bass,owilliams@example.com,"00447 Danny Overpass, Lunabury, WV 33833",9558316066,data202,84,B
163	Brenda,Hunter,sullivanhaley@example.net,"99531 Tamara Course, Derrikside, NV 44464",9822767017,data226,84,B
164	Daniel,Moore,amber34@example.com,"7915 Christian Island, Hansonstad, FR 26127",8179916985,data220,83,B
165	Benjamin,Oconnor,surt75@example.net,"8422 Turner View Apt. 852, Port Stacy, AK 98374",7195207475,data220,83,B
166	Kevin,Estrada,candice22@example.org,"953 Page Lodge Suite 218, New Tammy, FW 98698",8296519572,data226,83,B
167	Adam,Phillips,ecarter@example.net,"Unit 6347 Box 4074, DPO AE 62072",9937070077,data220,83,B
168	Joseph,Benton,jennifer12@example.com,"1696 Blanchard Valleys, Williamsborough, TX 82406",6752287834,data228,83,B
169	Sandra,Harper,jeremygarrett@example.com,"4735 Justin Freeway, New Jay, VI 09859",6986781701,data202,83,B
170	James,Campos,kaitlyn85@example.net,"574 Larsen View, West Jeremyland, SC 03652",6011685870,data202,83,B
171	Walter,Wright,nicolelewis@example.net,"7873 Brett Flat, West Margaretchester, AL 90327",9411146917,data230,83,B
172	Samantha,Collins,renee12@example.net,"15320 Diana Expressway Apt. 605, Jacksonchester, NC 91229",7190600777,data245,83,B
173	James,Murphy,brandonphillips@example.com,"444 Joshua Cape Apt. 091, Masonburgh, CA 02052",6329632198,data255,82,B

Student records.csv after modified grade lookup

```

login_system.py student_system.py main.py course_system.py grade_system.py faculty_system.py x test.py
faculty_system.py > Faculty > get_grade_lookup
6 class Faculty:
442 def save_updated_data(self, students):
443     with open(self.student_records, mode='w', newline='') as file:
444         fieldnames = ['first_name', 'last_name', 'email', 'address', 'phone_number', 'course_id', 'marks', 'grade']
445         writer = csv.DictWriter(file, fieldnames=fieldnames)
446         writer.writeheader()
447         for student in students:
448             writer.writerow(student)
449         print(f"Student records updated and saved to {self.student_records}.")

```

- save_grade_lookup_to_file(): saves the modified grade lookup to *grades.csv*

1	grades,maximum,minimum
2	A+,100,80
3	A,79,70
4	B,69,60
5	C,59,50
6	D,49,40
7	E,39,35
8	F,34,0
9	

Original grade lookup stored in grades.csv

1	grades,maximum,minimum
2	A+,100,95
3	A,94,90
4	B,89,80
5	C,79,70
6	D,69,60
7	E,59,50
8	F,49,0
9	

Modified grade lookup

```
login_system.py student_system.py main.py course_system.py grade_system.py faculty_system.py X
faculty_system.py > Faculty > get_grade_lookup
6 class Faculty:
450 def save_grade_lookup_to_file(self, grade_lookup):
451     grade_order = ['A+', 'A', 'B', 'C', 'D', 'E', 'F']
452     grade_lookup_sorted = sorted(grade_lookup, key=lambda x: grade_order.index(x['grade']))
453     with open(self.grade_file, mode='w', newline='') as file:
454         fieldnames = ['grades', 'maximum', 'minimum']
455         writer = csv.DictWriter(file, fieldnames=fieldnames)
456         writer.writeheader()
457         for grade in grade_lookup_sorted:
458             writer.writerow({
459                 'grades': grade['grade'],
460                 'maximum': grade['maximum'],
461                 'minimum': grade['minimum']
462             })
463     print(f"Modified grade lookup saved to '{self.grade_file}' in the correct order.")
```

- delete_faculty_account(): deletes the faculty account based on the email entered

```
grades.csv course_records.csv faculty_data.csv login.csv X
1032 danny.murphy@sjsu.edu,123456,professor
1033 matthew.jones@sjsu.edu,123456,professor
1034 mary.sullivan@sjsu.edu,123456,professor
1035 andrew.perez@sjsu.edu,123456,professor
1036 colin.espinoza@sjsu.edu,123456,professor
1037 stephen.smith@sjsu.edu,123456,professor
1038 gary.harmon@sjsu.edu,123456,professor
1039 joshua.harris@sjsu.edu,123456,professor
1040 sarah.russell@sjsu.edu,123456,professor
1041 daniel.jackson@sjsu.edu,123456,professor
1042 emily.dalton@sjsu.edu,123456,professor
1043 mark.miller@sjsu.edu,123456,professor
1044 leah.phillips@sjsu.edu,123456,professor
1045 frederick.johnson@sjsu.edu,123456,professor
1046 heather.bailey@sjsu.edu,123456,professor
1047 carol.wright@sjsu.edu,123456,professor
1048 andrew.mills@sjsu.edu,123456,professor
1049 christy.swanson@sjsu.edu,123456,professor
1050 deborah.humphrey@sjsu.edu,123456,professor
1051 jose.rush@sjsu.edu,123456,professor
1052 tiffany.tran@sjsu.edu,123456,professor
1053 laura.long@sjsu.edu,123456,professor
1054 laura.solis@sjsu.edu,123456,professor
1055 patrick.soto@sjsu.edu,123456,professor
1056 rachel.zane@sjsu.edu,1jqhtrj,student
1057
```

Professor 'sheldon.cooper@sjsu.edu' deleted from login.csv

```
grades.csv course_records.csv faculty_data.csv login.csv student_records.csv new 2 new
35 Andrew, Perez, andrew.perez@sjsu.edu, Rasmussenborough, 3805059333, data245, assistant professor
36 Colin, Espinoza, colin.espinoza@sjsu.edu, Port Kevinberg, 4307455331, data245, junior professor
37 Stephen, Smith, stephen.smith@sjsu.edu, New Anthony, 4660878874, data245, head of department
38 Gary, Harmon, gary.harmon@sjsu.edu, Moorefurt, 7617185591, data228, assistant professor
39 Joshua, Harris, joshua.harris@sjsu.edu, East Shane, 6331123628, data228, assistant professor
40 Sarah, Russell, sarah.russell@sjsu.edu, Tapiastad, 0315915949, data228, assistant professor
41 Daniel, Jackson, daniel.jackson@sjsu.edu, East Gabriel, 6559226722, data228, assistant professor
42 Emily, Dalton, emily.dalton@sjsu.edu, West Lisa, 7590002851, data228, junior professor
43 Mark, Miller, mark.miller@sjsu.edu, Kevinmouth, 9447349982, data228, head of department
44 Leah, Phillips, leah.phillips@sjsu.edu, Michaelhaven, 5554400642, data230, assistant professor
45 Frederick, Johnson, frederick.johnson@sjsu.edu, Brownhaven, 4259756656, data230, assistant professor
46 Heather, Bailey, heather.bailey@sjsu.edu, Valeriestad, 5051123202, data230, assistant professor
47 Carol, Wright, carol.wright@sjsu.edu, Robinsonton, 4872410640, data230, assistant professor
48 Andrew, Mills, andrew.mills@sjsu.edu, Port Brettside, 1714957578, data230, junior professor
49 Christy, Swanson, christy.swanson@sjsu.edu, West Stephanie, 9444550354, data230, head of department
50 Deborah, Humphrey, deborah.humphrey@sjsu.edu, West Danielmouth, 7111465926, data255, assistant professor
51 Jose, Rush, jose.rush@sjsu.edu, Youngmouth, 5032036259, data255, assistant professor
52 Tiffany, Tran, tiffany.tran@sjsu.edu, Robinsonmouth, 9276036313, data255, assistant professor
53 Laura, Long, laura.long@sjsu.edu, North Jenniferstad, 6050705360, data255, assistant professor
54 Laura, Solis, laura.solis@sjsu.edu, South David, 4872369563, data255, junior professor
55 Patrick, Soto, patrick.soto@sjsu.edu, East Elijah, 4286177622, data255, head of department
56
```

Professor 'sheldon.cooper@sjsu.edu' deleted from faculty_data.csv


```
login_system.py student_system.py main.py course_system.py grade_system.py faculty_system.py X
faculty_system.py > Faculty > get_grade_lookup
6 class Faculty:
464 def delete_faculty_account(self, email):
465     rows = []
466     with open(self.faculty_file, mode = 'r') as file:
467         reader = csv.reader(file)
468         rows = list(reader)
469         initial_row_count = len(rows)
470         rows = [row for row in rows if row[2] != email]
471         if len(rows) < initial_row_count:
472             with open(self.faculty_file, mode = 'w', newline = '') as file:
473                 writer = csv.writer(file)
474                 writer.writerows(rows)
475             print(f"Faculty account with email {email} has been deleted from faculty records.")
476         else:
477             print(f"No faculty records found for email {email} in faculty records.")
478             login_rows = []
479             with open(self.login_file, mode = 'r') as file:
480                 reader = csv.reader(file)
481                 login_rows = list(reader)
482                 login_row_count = len(login_rows)
483                 login_rows = [row for row in login_rows if row[0] != email or row[2] != "professor"]
484                 if len(login_rows) < login_row_count:
485                     with open(self.login_file, mode = 'w', newline = '') as file:
486                         writer = csv.writer(file)
487                         writer.writerows(login_rows)
488                     print(f"Faculty account with email {email} has been deleted from login system.")
489                 else:
490                     print(f"No login record found for email {email} in the login system.")
491             ...

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
12. Delete Course
13. Modify Grade Lookup
14. Delete your account
15. Change password
16. Return to main menu
Please choose an option (1/2/3/4/5/6/7/8/9/10/11/12/13/14/15/16): 14
Enter your email to delete account: sheldon.cooper@sjsu.edu
Faculty account with email sheldon.cooper@sjsu.edu has been deleted from faculty records.
Faculty account with email sheldon.cooper@sjsu.edu has been deleted from login system.
```

Console Output

- `sort_student_records()`: This follows a menu driven approach and enables user to choose whether they want to sort student records by email or by marks.

grades.csv	course_records.csv	faculty_data.csv	login.csv	student_records.csv	new 2	new 1	test_course_records.csv
1 first_name,last_name,email,address,phone_number,course_id,marks,grade							
2 Michael,Myers,joseph44@example.net,"2390 Stephanie Fork Apt. 729, North Jenniferborough, TX 83960",6759515454,data226,87,B							
3 Cindy,Collins,nvillanueva@example.org,"8532 Garcia Dale, Martinezview, NJ 60678",9795557828,data228,87,B							
4 Andrew,Williams,thomasaahley@example.com,"0857 Heidi Cape, Dillonport, KY 94133",8548622994,data201,87,B							
5 Erin,Walker,amber75@example.net,"2718 Mark Shoal, Susanstad, WV 45567",9083915684,data246,86,B							
6 Carrie,Lamb,craig@example.com,"4493 Avery Route Suite 852, Port Jenniferfurt, NH 210497,9881516853,data245,100,A+							
7 Kimberly,Hause,darrellhayes@example.org,"PSC 1123, Box 5747, APO AA 69805",7038558362,data230,100,A+							
8 Julie,Collins,hudsonamy@example.com,"FSC 0820, Box 7069, APO AE 79293",8041251551,data201,100,A+							
9 Samuel,Clark,jeffreyyoung@example.com,"32588 Jennifer Mill, Sherryfort, MD 52529",9471432742,data220,100,A+							
10 Elizabeth,Campbell,michelle25@example.com,"359 Callahan Inlet Apt. 439, Rebekahfurt, WY 86635",7952899509,data220,100,A+							
11 Katelyn,Perkins,ryanrhodes@example.org,"Unit 9833 Box 6048, DPO AE 24948",8985496916,data245,100,A+							
12 Timothy,Peterson,webberic@example.com,"769 Thompson Drive Suite 920, South Justinbury, WI 14365",9802708019,data202,100,A+							
13 Kelly,Cruz,sperry@example.org,"927 Rose Orchard Apt. 627, Luasaton, MA 47681",8203279387,data220,100,A+							
14 Joseph,Hayes,brendapayne@example.org,"Unit 9258 Box 4192, DPO AA 64529",9173425582,data202,99,A+							
15 Kristen,Yoder,james56@example.org,"5187 Tamara Station, Bradleyfort, WV 41270",8649108020,data226,99,A+							
16 Neil,Mann,osolis@example.org,"32285 Beck Village Apt. 754, South Tina, KY 91584",7468209702,data220,99,A+							
17 Rachel,Jenkins,sheltonwilliam@example.com,"5206 Jenkins Place Suite 027, Jimenezshire, FR 84267",8903410550,data230,99,A+							
18 Teresa,Hayes,tina57@example.net,"52578 Audrey Freeway, Acostaberg, CO 44377",7693638675,data230,99,A+							
19 Alexander,Burke,watsonmorgan@example.net,"169 Marie Points, East Austin, MO 42597",9633726927,data230,99,A+							
20 Hannah,Fox,xbowen@example.com,"2563 John Loop, Roberthaven, VA 09956",5591270215,data200,99,A+							
21 Adam,Rodriguez,yolanda02@example.com,"64002 Jamie Crossing, New Marktown, CT 39163",7895079444,data228,99,A+							
22 Anna,Davis,alexistephens@example.org,"695 Joseph Valleys, South Jennifer, MD 19637",8929945376,data202,98,A+							
23 Kristen,Garcia,bakercarl@example.com,"971 Victor Islands Apt. 056, Williamburgh, WY 29349",7328816566,data220,98,A+							
24 James,Morales,christinacannon@example.org,"45693 Emily Brook Suite 794, Ryantown, ME 61893",6587053036,data202,98,A+							
25 Ann,Hodges,collinsgary@example.net,"27684 Fleming Gardens Suite 031, Port Kevinburgh, MD 02004",8486930917,data220,98,A+							
26 Kathleen,Brown,danielmoherson@example.net,"1620 Chen Brooks Suite 040, Carliside, LA 22433",9876793511,data245,98,A+							
27 Jared,Green,elizabethandrews@example.net,"Unit 5359 Box 4044, DPO AA 57526",6363392991,data245,98,A+							
28 Morgan,Johnson,ethomas@example.com,"722 Jennifer Wall Apt. 358, North Derekmouth, DE 66415",5184952288,data228,98,A+							
29 Jonathon,Butler,harpersstephanie@example.org,"5019 Margaret Walk, Lake Daisyburgh, ME 90918",7918194868,data245,98,A+							
30 Teresa,Fowell,jillharris@example.org,"0711 Valdez Unions Apt. 992, East Jessicaport, LA 77202",8599919259,data255,100,A+							
31 Brad,Collins,kimberlysuarez@example.com,"6428 Thomas Crescent, Caldwelltown, MS 13875",8682950463,data230,100,A+							
32 Elizabeth,Potter,kurt25@example.org,"2183 Susan Turnpike, Lake Danielle, RI 08010",8456492426,data255,100,A+							
33 Andrea,Mosley,vhanna@example.net,"USNS Ford, FPO AP 85256",6920496512,data245,98,A+							
34 Joseph,Alvarez,dwilson@example.net,"19750 Moreno Orchard, New Maryview, VT 60764",9188369049,data230,97,A+							
35 Angela,Orozco,lwoods@example.org,"9643 Joseph View, Port Shaneview, VA 58932",8245392574,data230,97,A+							
36 Patrick,Diaz,wesley0@example.net,"5344 Clark Circle, East Samuel, OR 40142",7865944464,data220,97,A+							
37 Zachary,Bolton,fetokex@example.net,"9605 Barnett Spur Apt. 023, North Brian, NY 69757",6654942409,data200,96,A+							
38 Kevin,Lynch,johnjones@example.com,"7831 Ortiz Avenue, Kirbyborough, RI 70085",6405559791,data245,96,A+							
39 Hector,May,lrobertson@example.net,"009 Samantha Fort, Port Brittany, VA 77605",7194597236,data202,96,A+							
40 Rebekah,Allen,pachecokaren@example.net,"2953 Buchanan Crossing Suite 827, Catherineside, DC 66474",7984594963,data245,96,A+							
41 Tara,Archer,ramirezstephanie@example.com,"38496 Cruz Harbor, Ashleyland, NV 41868",8280804103,data245,96,A+							
42 Shannon,Johnson,rosesherman@example.org,"070 Reed Alley Suite 970, New John, CO 72880",9698328955,data220,96,A+							
43 Tina,Le,steachicks@example.org,"115 Candice Club Suite 752, Carlssonview, MA 96930",7316141933,data230,96,A+							
44 Audrey,Ochoa,terrancecox@example.net,"5527 Arnold Lane Suite 690, East Christopher, SD 688457,7259753171,data245,96,A+							
45 Charles,Phillips,warrenbrooke@example.com,"051 Sara Junction Apt. 671, Port James, AS 15871",8713278049,data226,96,A+							
46 Marie,White,howardharris@example.com,"49273 Monica Bypass, Kevinmouth, NE 22367",6349842731,data245,95,A+							
47 Tracy,Fuller,mkenney@example.org,"FSC 5792, Box 1383, APO AE 08541",7645862693,data245,95,A+							
48 Cassandra,Perry,ramos@example.com,"51347 Jon Village, Lake Melissa, SC 22833",6012700948,data202,98,A+							
49 Jennifer,Foster,melissapollard@example.net,"4581 Chase Corner Apt. 714, South Tiffanyview, IN 65041",8598724080,data245,98,A+							
50 Sarah,Jones,michaelmartin@example.net,"150 Baker Mission Apt. 213, Kimpport, ID 69894",7445077003,data230,98,A+							
51 James,Williams,dan@example.net,"521 Foster Crest, West Robertchester, SC 60259",6097591503,data230,98,A+							
52 Tamara,Chen,rhodesmegan@example.org,"5527 Michael Parkways, West Emilyton, MD 13309",9405182621,data230,98,A+							
53 Danielle,Lloyd,ghines@example.org,"5903 Amy Branch, South John, MT 78094",7398672607,data230,95,A+							
54 Wendell,Burton,scottbaird@example.com,"86458 Gary Hill, New Steady, MD 566588,6582803880,data230,95,A+							

Original student_records.csv

	grades.csv	course_records.csv	faculty_data.csv	login.csv	student_records.csv	new 2	new 1	test_course_records.csv
1					first_name,last_name,email,address,phone_number,course_id,marks,grade			
2					Judith,Anthony,aaron08@example.net,"6664 Chen Canyon Suite 006, Troymouth, CA 06938",752212221,data202,22,F			
3					Brittany,Wilcox,abanks@example.org,"47131 Freeman Creek, Mannton, MI 91717",9467908188,data228,69,D			
4					Karen,Burgess,abauer@example.com,"PSC 6170, Box 0695, APO AA 99179",7386350992,data230,31,F			
5					Anthony,Ramirez,ablackburn@example.net,"FSC 4442, Box 9029, APO AA 37956",6131374734,data202,20,F			
6					Teresa,Tapia,achan@example.net,"USS Kennedy, FPO AE 55849",6886867662,data201,72,C			
7					Hayley,Stewart,acox@example.org,"1909 Shelton Cliff Suite 326, North Craig, WI 07682",9411986921,data245,71,C			
8					George,Scott,adamarios@example.net,"09679 Fisher Knolls, South Matthew, MP 97328",7598116849,data245,29,F			
9					Edward,Johnson,adamsandrew@example.com,"31586 Kevin Port Suite 446, Conleyborough, NY 51043",6566162493,data228,33,F			
10					Elizabeth,Baxter,adamajennifer@example.net,"628 Mitchell Parks, South Robertport, NV 49836",8520321601,data202,21,F			
11					Carrie,Turner,adamsherry@example.net,"81318 Julia Road Suite 406, South Curtis, NJ 41428",8446246803,data200,91,A			
12					Zachary,Grimes,afisher@example.com,"7187 Arias Key, West Steven, NE 29378",6975984284,data245,39,F			
13					Angela,Hughes,ashorne@example.net,"282 Jordan Highway Suite 344, New Sandrville, OR 64603",8315899945,data245,35,F			
14					Charles,Reeves,alexandria01@example.com,"1327 Knight Field, Jeffreychester, FW 55201",7186189546,data226,4,F			
15					Ryan,Stewart,alexandria12@example.net,"9885 Johnson Station Suite 550, West Eric, NY 44226",6191883373,data230,31,F			
16					Anna,Davis,alexisstephens@example.org,"695 Joseph Valleys, South Jennifer, MD 19637",8929945376,data202,98,A+			
17					Elizabeth,Burton,ali@example.org,"PSC 9900, Box 1801, APO AE 84544",8483873585,data220,37,F			
18					Krystal,Lewis,allisonalhoun@example.net,"9976 Martha Knoll Suite 248, Port Shawn, WI 35036",7796503359,data245,22,F			
19					Rebecca,Miller,allisonlong@example.com,"96380 Erika Stravenue, Williamberg, UT 29666",6122339810,data245,1,F			
20					Peter,Boyle,amandaharris@example.com,"061 White Highway, South George, KS 33169",8503147194,data202,6,F			
21					Brooke,Nunez,amandamartin@example.net,"PSC 3584, Box 0713, APO AA 79056",9641113875,data230,46,F			
22					Daniel,Moore,amber34@example.com,"7915 Christian Island, Hansonstad, FR 26127",8179916965,data220,83,B			
23					Margaret,Barry,amber76@example.com,"12677 Hawkins Villa, South Jean, FR 66429",8969731426,data202,72,C			
24					Erin,Walker,amber76@example.net,"2718 Mark Shoal, Susanstad, WY 45567",9083915684,data226,36,B			
25					Paul,Chen,amorrisson@example.org,"5893 Rogers Fords Suite 120, North Michaeland, AS 89148",6561662010,data200,9,F			
26					Haley,Delgado,amy87@example.org,"4916 Lauren Ports Suite 404, New Natalie, LA 34196",7218010375,data201,24,F			
27					Ryan,Gordon,amylndry@example.com,"7467 Cantu Key, Brianshire, NE 60859",7122968695,data230,46,F			
28					David,Walker,ana95@example.net,"9760 Cynthia Branch, Lake Timothymouth, NC 32406",9017327556,data202,71,C			
29					Christian,Johnson,andersenbrian@example.org,"14141 Christopher Way Suite 817, Davidbury, NJ 24035",9053212216,data202,17,F			
30					Autumn,Sanchez,andersencharles@example.net,"803 Martin Key Suite 555, Krauseville, WA 78771",8082935000,data200,29,F			
31					Elaine,Perer,anderseandanielle@example.org,"776 Kimberly Islands Suite 320, Scottville, AS 52737",8223656985,data201,93,A			
32					Barbara,Sharp,andersejennifer@example.com,"6212 Burgess Gateway Suite 573, East Daniel, AZ 52235",8994714305,data255,73,C			
33					Edward,Miller,andreww02@example.net,"052 Jacob Turnpike Apt. 743, Hernandezchester, FM 67627",9773732923,data220,34,F			
34					Patricia,Rodriguez,andreww42@example.org,"503 Cohen Turnpike, Mooreville, MH 95009",6525485189,data200,84,B			
35					Brittany,Thompson,andreww56@example.net,"19981 Riskey Mission, Lake Ryan, DC 96439",9084881828,data226,68,D			
36					Aaron,Gilbert,andrewworsey@example.org,"92405 Michael Centers Suite 798, West Brandon, VT 78479",9967160430,data255,18,F			
37					Michelle,Jones,angela7@example.net,"11793 Matthew Divide Suite 826, Aliciport, NV 32287",6044780154,data202,90,A			
38					Destiny,Garrett,angela7@example.net,"62793 Timothy Lane Suite 252, Ashleyport, NC 67946",7006315630,data255,4,F			
39					Anthony,Boyd,angela79@example.net,"9854 George Lights Suite 469, Johnsonfurt, MN 78505",8103478321,data255,27,F			
40					James,Sanchez,angela86@example.com,"03214 Stokes Courts Suite 605, Davisport, RI 95406",9964566484,data200,37,F			
41					Keith,Hopkins,angelabarber@example.org,"2779 Daniel Square Apt. 642, Andreaport, NE 98029",7746939750,data228,65,D			
42					Darin,Johnson,angelashicks@example.org,"97546 Diaz Skyway, South Jennifer, FR 59103",6376522722,data245,1,F			
43					James,Villa,angela8jones@example.org,"19650 Davis Hills, West Christopher, SD 21568",9869249785,data202,75,C			
44					Anne,Adams,anitaunderwood@example.net,"1859 Stokes Rapid, North Dana, SD 10623",8770932214,data202,57,E			
45					Christopher,Johnson,anna03@example.com,"1485 Thomas Corner, South Paulashire, OR 15990",8290214255,data201,17,F			
46					Michael,Hickman,annerodriguez@example.com,"361 Brandon Wall Suite 590, Port Andrew, MD 17439",8432309395,data201,59,E			
47					James,Avery,annette97@example.net,"7280 Rachel Knolls Suite 337, East David, IL 47809",9120266999,data230,43,F			
48					Allison,Lam,annette8vidason@example.com,"5490 Brittany Motorway Apt. 935, West Anna, VT 71436",9273430632,data201,92,A			
49					Michael,Lee,annleonard@example.org,"0757 Benjamin Hollow, New Charles, VT 74354",8488522297,data230,23,F			
50					Brenda,Davis,anthony82@example.org,"6359 Brown Circle Suite 833, Readton, NY 50446",6801966395,data220,64,D			
51					Sandra,Cook,apearson@example.org,"9191 Briana Junction, Sanchezland, FR 76298",7781451882,data255,66,D			
52					Andre,Martinez,april13@example.com,"8514 Carter Place, Patelborough, OR 20345",6633466948,data220,12,F			
53					Mark,Tate,arnoldbrett@example.com,"1866 Kristi Greens Suite 135, South Scott, UT 30627",9904637568,data226,15,F			
54					Stephanie,Lynch,arrell42@example.net,"8651 Stokes Manor, North David, MP 58028",6087832620,data245,1,F			

student_records.csv when sorted by email

	grades.csv	course_records.csv	faculty_data.csv	login.csv	student_records.csv	new 2	new 1	test_course_records.csv
1					first_name,last_name,email,address,phone_number,course_id,marks,grade			
2					Carrie,Lamb,coraig@example.com,"4433 Avery Route Suite 852, Port Jenniferfurt, NH 21049",9981516853,data245,100,A+			
3					Kimberly,Houze,darrellhayes@example.org,"PSC 1123, Box 5747, APO AA 69805",7038558362,data230,100,A+			
4					Julie,Collins,hudsonamy@example.com,"PSC 0820, Box 7069, APO AE 79293",8041251551,data201,100,A+			
5					Samuel,Clark,jeffreyyoung@example.com,"32588 Jennifer Mill, Sherryfort, MD 52529",9471432742,data220,100,A+			
6					Teresa,Powell,jillharris@example.org,"0711 Valdez Unions Apt. 992, East Jessicaport, LA 77202",8599919259,data255,100,A+			
7					Brad,Collins,kimberlysuarez@example.com,"6428 Thomas Crescent, Caldwelltown, MS 13875",8682950463,data230,100,A+			
8					Elizabeth,Potter,kurt25@example.org,"2183 Susan Turnpike, Lake Danielle, RI 08010",8456492426,data255,100,A+			
9					Elizabeth,Campbell,michelle25@example.com,"359 Callahan Inlet Apt. 439, Rebekahfurt, WY 86635",7952899509,data220,100,A+			
10					Rachel,Zane,rachel.zane@sjsu.edu,San Jose,9432600435,data202,100,A+			
11					Katelyn,Perkins,ryanrhodes@example.org,"Unit 9833 Box 6048, DPO AE 24948",8988496916,data245,100,A+			
12					Timothy,Peterson,webberic@example.com,"769 Thompson Drive Suite 920, South Justinbury, WI 14365",9802708019,data202,100,A+			
13					Kelly,Cruz,xperry@example.org,"927 Ross Orchard Apt. 627, Lucaston, MA 47681",8203279387,data220,100,A+			
14					Joseph,Hayes,brendapayne@example.org,"Unit 9258 Box 4192, DPO AA 64529",9173425582,data202,99,A+			
15					Kristen,Yoder,james56@example.org,"5187 Tamara Station, Bradleyfort, WV 41270",8649108020,data226,99,A+			
16					Neil,Mann,osolis@example.org,"32285 Beck Village Apt. 754, South Tina, KY 91584",7468209702,data220,99,A+			
17					Rachel,Jenkins,sheltonwilliam@example.com,"5206 Jenkins Place Suite 027, Jimenezshire, FR 84267",8903410550,data230,99,A+			
18					Teresa,Hayes,tina57@example.net,"52578 Audrey Freeway, Acostaberg, CO 44377",7693638675,data230,99,A+			
19					Alexander,Burke,watsonmorgan@example.net,"169 Marie Points, East Austin, MO 42597",9633726927,data230,99,A+			
20					Hannah,Fox,xhoben@example.com,"2563 John Loop, Roberthaven, VA 09956",8591270215,data200,99,A+			
21					Adam,Rodriguez,yolanda02@example.com,"64002 Jamie Crossing, New Marktown, CT 39163",7895079444,data228,99,A+			
22					Anna,Davis,alexisstephens@example.org,"695 Joseph Valleys, South Jennifer, MD 19637",8929945376,data202,98,A+			
23					Kristen,Garcia,bakercarl@example.com,"971 Victor Islands Apt. 056, Williamburgh, WY 29349",7328816566,data220,98,A+			
24					James,Morales,christinacannon@example.org,"45693 Emily Brook Suite 794, Ryantown, ME 61893",6587053036,data202,98,A+			
25					Ann,Hodges,collinsgary@example.net,"27684 Fleming Gardens Suite 031, Port Kevinburgh, MD 02004",8486930917,data220,98,A+			
26					Kathleen,Brown,danielmcperson@example.net,"1620 Chen Brooks Suite 040, Carlside, LA 22433",9876793511,data245,98,A+			
27					Jared,Green,elizabethandrews@example.net,"Unit 5359 Box 4044, DPO AA 57526",6363392991,data245,98,A+			
28					Morgan,Johnson,ethomas@example.com,"722 Jennifer Wall Apt. 358, North Derekmouth, DE 66415",8184932888,data228,98,A+			
29					Jonathon,Butler,harperstephanie@example.org,"8019 Margaret Walk, Lake Daisyburgh, ME 90918",7918194868,data245,98,A+			
30					Cassandra,Perry,iramos@example.com,"51347 Jon Village, Lake Melissa, SC 22833",6012700948,data202,98,A+			
31					Jennifer,Foster,melissapollard@example.net,"4581 Chase Corner Apt. 714, South Tiffanyview, IN 65041",8598724080,data245,98,A+			
32					Sarah,Jones,michaelmartin@example.net,"150 Baker Mission Apt. 213, Kimport, ID 69994",7445077003,data230,98,A+			
33					James,Williams,osteeie@example.net,"521 Foster Crest, West Robertchester, SC 60259",6097591508,data230,98,A+			
34					Tamara,Chen,rhodesmegan@example.org,"5527 Michael Parkways, West Emilyton, ND 13309",9405182621,data230,98,A+			
35					Andrea,Mosley,vhanna@example.net,"USNS Ford, FPO AP 85256",6920496512,data245,99,A+			
36					Joseph,Alvarez,dwilson@example.net,"19750 Moreno Orchard, New Maryview, VT 60764",9188369049,data230,97,A+			
37					Angela,Orozco,lwoods@example.org,"9643 Joseph View, Port Shaneview, WA 58932",8245392574,data230,97,A+			
38					Patrick,Diaz,wanda09@example.net,"5344 Clark Circle, East Samuel, OR 40162",7865944448,data220,97,A+			
39					Zachary,Bolton,estokes@example.net,"9605 Barnett Spur Apt. 023, North Brian, NY 69757",6654942409,data200,96,A+			
40					Kevin,Lynch,johnjones@example.com,"7831 Ortiz Avenue, Kirbyborough, RI 70085",6405559791,data245,96,A+			
41					Hector,May,lobertson@example.net,"009 Samantha Fort, Port Brittany, VA 77605",7194597236,data202,96,A+			
42					Rebekah,Allen,pachecokaren@example.net,"2953 Buchanan Crossing Suite 827, Catherineside, DC 66474",7984594963,data245,96,A+			
43					Tara,Archer,ramirezstephanie@example.com,"38496 Cruz Harbor, Ashleyland, NV 41868",8280804103,data245,96,A+			
44					Shannon,Johnson,roshermerman@example.org,"070 Reed Alley Suite 970, New John, CO 72880",9698328955,data220,96,A+			
45					Tina,Le,stacychicks@example.org,"115 Candice Club Suite 752, Carlsonview, MA 96930",7316141933,data230,96,A+			
46					Audrey,Ochoa,terrancecox@example.net,"9877 Arnold Lane Suite 690, East Christopher, SD 68845",7259753171,data245,96,A+			
47					Charles,Phillips,warrenbrooke@example.com,"051 Sara Junction Apt. 671, Port James, AS 15871",8713278049,data226,96,A+			
48					Marie,White,howardharris@example.com,"49273 Monica Bypass, Kevinmouth, NE 22367",6349842731,data245,95,A+			
49					Tracy,Fuller,mkenney@example.org,"PSC 5792, Box 1383, APO AE 08541",7645862693,data245,95,A+			
50					Danielle,Lloyd,ghines@example.org,"8903 Amy Branch, South John, MT 78094",7398672607,data230,95,A+			
51					Harold,Burton,rachelhoward@example.com,"8455 Carr Hill, New Stacy, MP 36585",6068029389,data220,95,A+			
52					Elaine,Lang,rmoon@example.net,"452 Thomas Locks Suite 974, Petermouth, FL 01474",8023758664,data220,95,A+			
53					Rick,James,robert65@example.org,"662 Walker Mount, Ronnieport, CA 82643",8434352395,data226,95,A+			
54					Tina,Stevens,ronald3@example.com,"50188 Tanya Creek, East Michaelstad, AZ 30468",7086560080,data201,95,A+			

student_records.csv when sorted by marks

```

login_system.py student_system.py main.py course_system.py grade_system.py faculty_system.py X
faculty_system.py > Faculty > delete_faculty_account
6 class Faculty:
491     def sort_student_records(self):
492         print("Sort the students records by:")
493         print("1. Email")
494         print("2. Marks")
495         choice = input("Enter your choice (1 or 2): ")
496         try:
497             with open(self.student_records, mode = 'r') as file:
498                 reader = csv.reader(file)
499                 header = next(reader)
500                 student_records = list(reader)
501         except FileNotFoundError:
502             print(f"Error: The file '{self.student_records}' was not found.")
503             return
504         start_time = time.time()
505         if choice == '1':
506             student_records.sort(key=lambda x: x[2].lower())
507         elif choice == '2':
508             student_records.sort(key=lambda x: int(x[6]), reverse=True)
509         else:
510             print("invalid choice. Please enter 1 or 2")
511             return
512         end_time = time.time()
513         elapsed_time = end_time - start_time
514         print(f"\nSorting took {elapsed_time:.4f} seconds.")
515         try:
516             with open(self.student_records, mode='w', newline='') as file:
517                 writer = csv.writer(file)
518                 writer.writerow(header)
519                 for student in student_records:
520                     writer.writerow(student)
521         except:
522             print("Error: Could not save the sorted records.")
523             return
524         print("Student records have been sorted and saved.")

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

Sort the students records by:
1. Email
2. Marks
Enter your choice (1 or 2): 1

Sorting took 0.0004 seconds.

Student records have been sorted and saved.

```

Console Output

- `search_student_by_email()`: takes email as an input to search the student in `student_records.csv`.

```

login_system.py X student_system.py main.py course_system.py grade_system.py faculty_system.py
login_system.py > Login > main_menu
4 class Login:
121     def professor_menu(self, faculty, email):
155         elif choice == '6':
156             student_records = "C:/Users/aradh/Desktop/DATA 200 Python Programming/Lab1/student_records.csv"
157             grade_file = "C:/Users/aradh/Desktop/DATA 200 Python Programming/Lab1/grades.csv"
158             login_file = "C:/Users/aradh/Desktop/DATA 200 Python Programming/Lab1/login.csv"
159             course_file = "C:/Users/aradh/Desktop/DATA 200 Python Programming/Lab1/course_records.csv"
160             student = Student(student_records, grade_file, login_file, course_file)
161             email = input("Enter the email to search for student records: ")
162             import time
163             start_time = time.time()
164             student.search_student_by_email(email)
165             end_time = time.time()
166             elapsed_time = end_time - start_time
167             print(f"Time taken to search for the student record: {elapsed_time:.4f} seconds")

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

Enter the email to search for student records: rachel.zane@sjsu.edu
Student Record:
Name: Rachel Zane
Email: rachel.zane@sjsu.edu
Address: San Jose
Phone Number: 9432600435
Course ID: data202
Marks: 100
Grade: A+
-----
Course ID: data230
Marks: 45
Grade: F
-----

Total Marks: 145
Average Marks: 72.50
Time taken to search for the student record: 0.0030 seconds
Time taken to search for the student record: 0.0031 seconds

```

- `delete_student_account()`: deletes the records of a student from `student_records.csv` and login details from `login.csv`.

```
grades.csv  course_records.csv  faculty_data.csv  login.csv
1032 danny.murphy@sjsu.edu,123456,professor
1033 matthew.jones@sjsu.edu,123456,professor
1034 mary.sullivan@sjsu.edu,123456,professor
1035 andrew.perez@sjsu.edu,123456,professor
1036 colin.espinosa@sjsu.edu,123456,professor
1037 stephen.smith@sjsu.edu,123456,professor
1038 gary.harmon@sjsu.edu,123456,professor
1039 joshua.harris@sjsu.edu,123456,professor
1040 sarah.russell@sjsu.edu,123456,professor
1041 daniel.jackson@sjsu.edu,123456,professor
1042 emily.dalton@sjsu.edu,123456,professor
1043 mark.miller@sjsu.edu,123456,professor
1044 leah.phillips@sjsu.edu,123456,professor
1045 frederick.johnson@sjsu.edu,123456,professor
1046 heather.bailey@sjsu.edu,123456,professor
1047 carol.wright@sjsu.edu,123456,professor
1048 andrew.mills@sjsu.edu,123456,professor
1049 christy.swanson@sjsu.edu,123456,professor
1050 deborah.humphrey@sjsu.edu,123456,professor
1051 jose.rush@sjsu.edu,123456,professor
1052 tiffany.tran@sjsu.edu,123456,professor
1053 laura.long@sjsu.edu,123456,professor
1054 laura.solis@sjsu.edu,123456,professor
1055 patrick.soto@sjsu.edu,123456,professor
1056
```

Student 'rachel.zane@sjsu.edu' removed from login.csv

```
grades.csv  course_records.csv  faculty_data.csv  login.csv  student_records.csv  new 2  new 1  test_course_reco
995 Billy,Tucker,corysantana@example.com,"68570 Hogan Fork Apt. 117, Lake Joseph, NJ 68289",6191970140,data226,0,F
996 Nancy,Jones,dakotawelch@example.net,"6330 Faith Parks Suite 115, Lake Zacharyhaven, OK 70787",6378499636,data220,0,F
997 Mark,Baker,drodgers@example.net,"180 Jessica Parkways, West Kayla, GA 43584",6749721930,data228,0,F
998 Alex,Clay,graysean@example.org,"4699 Hampton Valleys Apt. 253, Williamshaven, PW 39382",8375955826,data245,0,F
999 Michael,Smith,janetnewman@example.org,"5103 Ronnie Green Suite 455, Clineland, IL 88679",7587695741,data255,0,F
1000 David,Hopkins,soniachandler@example.com,"16803 Spencer Estates, South Garytown, MP 06515",7881817363,data228,0,F
1001 Steven,Barry,valerierodriguez@example.com,"35671 Pittman River, Lake Jeanville, HI 73261",7129815440,data201,0,F
1002
```

Student 'rachel.zane@sjsu.edu' removed from student_records.csv

```
login_system.py X  student_system.py  main.py  course_system.py  grade_system.py  faculty_system.py
login_system.py > Login > professor_menu
4 class Login:
121 def professor_menu(self, faculty, email):
174 elif choice == '9':
175     student_records = "C:/Users/aradh/Desktop/DATA 200 Python Programming/Lab1/student_records.csv"
176     grade_file = "C:/Users/aradh/Desktop/DATA 200 Python Programming/Lab1/grades.csv"
177     login_file = "C:/Users/aradh/Desktop/DATA 200 Python Programming/Lab1/login.csv"
178     course_file = "C:/Users/aradh/Desktop/DATA 200 Python Programming/Lab1/course_records.csv"
179     student = Student(student_records, grade_file, login_file, course_file)
180     prof_email = input("Enter your email to delete a student account: ")
181     rank = faculty.get_professor_rank(prof_email)
182     if rank == "headofdepartment":
183         email = input("Enter the email of the student whose account you want to delete: ")
184         student.delete_student_account(email)
185     else:
186         print("Permission denied: Only the Head of Department can delete a student account.")

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
-----Sub Professor Menu-----
1. View your details
2. View professor list by Course ID
3. Modify your details
4. View Course Performance (sorted by average marks)
5. Sort student data by email or marks
6. Search student by email
7. Search professor by email
8. Modify student marks
9. Delete student account
10. View Course List
11. Add Course
12. Delete Course
13. Modify Grade Lookup
14. Delete your account
15. Change password
16. Return to main menu
Please choose an option (1/2/3/4/5/6/7/8/9/10/11/12/13/14/15/16): 9
Enter your email to delete a student account: patrick.soto@sjsu.edu
Enter the email of the student whose account you want to delete: rachel.zane@sjsu.edu
Student account with email rachel.zane@sjsu.edu has been deleted from student records.
Student account with email rachel.zane@sjsu.edu has been deleted from login system.
```

Console Output