project_main.py

```python
import os
import csv

class Person:
    def __init__(self, id, name, age, address) -> None:
        self.id = id
        self.name = name
        self.age = age
        self.address = address

        data = {
            "id": id,
            "name": self.name,
            "age": self.age,
            "address": self.address
        }

        try:
            # Check if file exists
            file_exists = os.path.isfile("person.csv")

            # Open the file in append mode and write data
            with open("person.csv", "a", newline="") as file:
                writer = csv.DictWriter(file, fieldnames=["id", "name", "age", "address"])

                if not file_exists:
                    writer.writeheader()

                writer.writerow(data)

        except Exception as e:
            print(f"An unexpected error occurred: {e}")

    @staticmethod
    def display_person_info(id):
        try:
            # Initialize variables to avoid 'referenced before assignment' error
            re_id = re_name = re_age = re_add = course_name = grade_num = None

            # Read the person.csv file
            with open("person.csv", "r") as pr:
                person_reader = csv.DictReader(pr)

                for p in person_reader:
                    if p["id"].strip() == id.strip():
                        re_id = p["id"]
                        re_name = p["name"]
                        re_age = p["age"]
                        re_add = p["address"]
                        break  # Exit loop once the person is found

            # Check if person data was found
            if not re_id:
                print("Person not found.")
                return

            # Read the course and grade files
            with open("en_course.csv", "r") as enc, open("course_grade.csv", "r") as gpa:
                course_reader = csv.DictReader(enc)
                grade_reader = csv.DictReader(gpa)

                # Find the enrolled course for the given ID
                for c in course_reader:
                    if c["id"].strip() == id.strip():
                        course_name = c["course"]
                        break  # Exit loop once the course is found

                # Check if course was found
                if not course_name:
                    print("Course not found.")
                    return

                # Find the grade for the course and ID
                for g in grade_reader:
                    if g["id"].strip() == id.strip() and g["course"].strip() == course_name:
                        grade_num = g["grade"]
                        break  # Exit loop once the grade is found

            # Display the information
            print(f"""
Student Information:
Name: {re_name}
ID: {re_id}
Age: {re_age}
Address: {re_add}
Enrolled Course: {course_name}
Grade: {{{course_name}: {grade_num}}}
""")

        except Exception as e:
            print(f"An error occurred: {e}")


class Student():
    # Class attributes to store grades and enrolled courses.
    grade_list = {}
    course_list = []

    def add_grade(self, sid, course, grade):
        try:
            # Open the files containing student and course data.
            with open("person.csv", "r") as pr, open("course.csv", "r") as cr:
                person_reader = csv.DictReader(pr)
                course_reader = csv.DictReader(cr)

                # Find the student by ID and print a confirmation message.
                for p in person_reader:
                    if p["id"].strip() == sid.strip():
                        print(f"Grade {grade} added for {p['name']} in ", end="")

                # Find the course by code and complete the message.
                for c in course_reader:
                    if c["Course Code"].strip() == course.strip():
                        print(f"{c['Course Name']}")

                        # Add the grade to the class's grade list.
                        key = c["Course Name"]
                        value = grade
                        g = {key: value}
                        self.grade_list.update(g)

            # Prepare data for saving in the course-grade file.
            course_grade = {
                "id": sid,
                "course": key,
                "grade": grade
            }

            # Append to 'course_grade.csv', creating a header if needed.
            file_exists = os.path.isfile("course_grade.csv")
            with open("course_grade.csv", "a", newline="") as file:
                writer = csv.DictWriter(file, fieldnames=["id", "course", "grade"])
                if not file_exists:
                    writer.writeheader()  # Add headers if the file is new.
                writer.writerow(course_grade)  # Add the new entry.

        except Exception as e:
            print(f"{e}")  # Handle and print any exceptions that occur.

    def enroll_course(self, sid, course):
        try:
            # Open the files containing student and course data.
            with open("person.csv", "r") as pr, open("course.csv", "r") as cr:
                person_reader = csv.DictReader(pr)
                course_reader = csv.DictReader(cr)

                # Find the student by ID and print a confirmation message.
                for p in person_reader:
                    if p["id"].strip() == sid.strip():
                        print(f"Student {p['name']} (ID: {p['id']}) ", end="")

                # Find the course by code and complete the message.
                for c in course_reader:
                    if c["Course Code"].strip() == course.strip():
                        course = c["Course Name"]
                        print(f"enrolled in {course}")
                        self.course_list.append(course)  # Store enrolled course.

            # Prepare data for saving in the enrolled courses file.
            en_course = {
                "id": sid,
                "course": course
            }

            # Append to 'en_course.csv', creating a header if needed.
            file_exists = os.path.isfile("en_course.csv")
            with open("en_course.csv", "a", newline="") as file:
                writer = csv.DictWriter(file, fieldnames=["id", "course"])
                if not file_exists:
                    writer.writeheader()  # Add headers if the file is new.
                writer.writerow(en_course)  # Add the new entry.

        except Exception as e:
            print(f"Unexpected error: {e}")  # Handle and print any exceptions.




class Course:
    def __init__(self, course_name, course_code, course_instructor) -> None:
        self.coursName = course_name
        self.coursCode = course_code
        self.coursInstructor = course_instructor
        self.student = []

        course_data = {
            "Course Name": self.coursName,
            "Course Code": self.coursCode,
            "Course Instructor": self.coursInstructor
        }

        try:
            # Check if file exists
            file_exists = os.path.isfile("course.csv")

            # Open the file in append mode and write data
            with open("course.csv", "a", newline="") as file:
                writer = csv.DictWriter(file, fieldnames=["Course Name", "Course Code", "Course Instructor"])

                if not file_exists:
                    writer.writeheader()

                writer.writerow(course_data)

        except Exception as e:
            print(f"An unexpected error occurred: {e}")


    def add_student():
        pass

    @staticmethod
    def display_course_info(course_code):
        try:
            # Initialize variables to avoid 'referenced before assignment' error
            cours_name = cours_code = cours_instructor = None

            # Read the course.csv file
            with open("course.csv", "r") as cr:
                course_read = csv.DictReader(cr)

                for c in course_read:
                    if c["Course Code"].strip() == course_code.strip():
                        cours_name = c["Course Name"]
                        cours_code = c["Course Code"]
                        cours_instructor = c["Course Instructor"]
                        break  # Exit loop once the person is found

            # Check if person data was found
            if not course_code:
                print("Course not found.")
                return

            # Read the course and grade files
            with open("en_course.csv", "r") as enc:
                course_reader = csv.DictReader(enc)

                # Find the enrolled course for the given ID
                for cr in course_reader:
                    if cr["course"] == cours_name:
                        student_id = cr["id"]
                        break  # Exit loop once the course is found

                # Check if Studen id was found
                if not student_id:
                    print("Student do not enrolled.")
                    return
            with open("person.csv", "r") as pr:
                reader = csv.DictReader(pr)

                for row in reader:
                    if row["id"].strip() == student_id.strip():
                        student_name = row["name"]

            # Display the information
            print(f"""
Course Information:
Course Name: {cours_name}
Course Code: {cours_code}
Instructor: {cours_instructor}
Enrolled Student: {student_name}

""")

        except Exception as e:
            print(f"An error occurred: {e}")

def main():
    print("""
==== Student Management System ====
1. Add New Student
2. Add New Course
3. Enroll Student in Course
4. Add Grade for Student
5. Display Student Details
6. Display Course Details
7. Save Data from File
8. Load Data from File
0. Exit
""")

    while True:
        try:
            option = int(input("Choose your option from above: "))
            if option < 0 or option > 8:
                print("Please choose a valid option (0-8).\n")
            elif option == 0:
                print("Exiting Student Management System. Goodbye!")
                break
            elif option == 1:
                name = input("Enter Name: ")
                age = input("Enter Age: ")
                address = input("Enter Address: ")
                std_id = input("Enter Student ID: ")

                Person(std_id, name, age, address)
                print(f"Student {name} (ID: {std_id}) added successfully.")
                break
            elif option == 2:
                cname = input("Enter Course Name: ")
                ccode = input("Enter Course Code: ")
                cinstructor = input("Enter Instructor Name: ")

                Course(cname, ccode, cinstructor)
                print(f"Course {cname} (Code: {ccode}) created with instructor {cinstructor}")
                break
            elif option == 3:
                std_id = input("Enter Student ID: ")
                ccode = input("Enter Course Code: ")
                student = Student()
                student.enroll_course(std_id, ccode)
                break
            elif option == 4:
                std_id = input("Enter Student ID: ")
                ccode = input("Enter Course Code: ")
                grade = input("Enter Grade: ")
                student = Student()
                student.add_grade(std_id,ccode,grade)
                break
            elif option == 5:
                std_id = input("Enter Student ID: ")
                Person.display_person_info(std_id)
                break
            elif option == 6:
                ccode = input("Enter Course Code: ")
                Course.display_course_info(ccode)
                break

        except ValueError:
            print("Invalid input! Please enter a number.")

if __name__ == "__main__":
    main()

```