AGUIRRE, PAUL VINCENT S.
M001-CP102-MW-4:00-06:30

DESCRIPTION:

The AttendanceSystem is a Python-based solution designed to manage employee attendance by recording clock-in and clock-out times using CSV files for storage. When the system starts, it first checks if the primary file (attendance.csv) exists. If not, it creates the file and writes the appropriate header row (including fields like Name, Time-in, Time-out, and Duration). It also initializes additional history files (edit_history.csv and delete_history.csv) intended to keep a log of any changes or deletions made later, ensuring that data modifications are traceable.

Once initialized, the system offers several core functionalities. For adding records, it prompts the user to enter their name. It then checks the existing records: if an entry exists for that name without a recorded clock-out time, the system interprets the action as clocking out, calculates the session's duration by determining the time difference from the recorded clock-in, and updates the record accordingly. If no such open record is found, the system records a new clock-in time.

In addition to adding records, users can list all existing attendance entries, which provides a comprehensive view of all clock-in and clock-out times along with calculated durations. For editing, the system allows a user to search for their records by name, choose a specific record, and update fields such as Name, Time-in, or Time-out. Any changes to time fields trigger a recalculation of the duration to ensure that the record reflects the correct elapsed time. Similarly, the deletion functionality enables a user to select and remove a specific record from the dataset.

The system operates through a menu-driven interface that continuously loops, prompting the user to choose an action from options such as clocking in/out, editing a record, listing records, deleting a record, or exiting the program. Each operation is accompanied by brief pauses to create a more user-friendly experience. Under the hood, the program leverages Python's built-in modules—csv for file operations, os for file management, time for creating delays, and datetime for timestamp handling—resulting in a robust yet straightforward attendance management system.

CODE:

```
import csv
import os
import time
import datetime

class AttendanceSystem:
    FILE_NAME = "attendance.csv"
    HISTORY_EDIT = "edit_history.csv"
    HISTORY_DELETE = "delete_history.csv"

    def __init__(self):
        self.init_file()

    def init_file(self):
        if not os.path.exists(self.FILE_NAME):
            print("Creating file...")
            time.sleep(2)
            print("File has been created.\n")
            time.sleep(2)

            with open(self.FILE_NAME, 'w', newline="") as f:
```

AGUIRRE, PAUL VINCENT S.
M001-CP102-MW-4:00-06:30

```
                writer = csv.DictWriter(f, fieldnames=["Name", "Time-in", "Time-out",
"Duration"])
                writer.writeheader()
        else:
            print("File already exists\n")
            time.sleep(1)

        for history_file in [self.HISTORY_EDIT, self.HISTORY_DELETE]:
            if not os.path.exists(history_file):
                with open(history_file, 'w', newline="") as f:
                    writer = csv.DictWriter(f, fieldnames=["Timestamp", "Name", "Field Edited",
"Old Value", "New Value"] if "edit" in history_file else ["Timestamp", "Name", "Deleted Record"])
                    writer.writeheader()

    def read_record(self):
        with open(self.FILE_NAME, 'r') as f:
            return list(csv.DictReader(f))

    def write_record(self, records):
        with open(self.FILE_NAME, 'w', newline="") as f:
            writer = csv.DictWriter(f, fieldnames=["Name", "Time-in", "Time-out", "Duration"])
            writer.writeheader()
            writer.writerows(records)

    def add_record(self):
        get_name = input("Enter name: ").strip().title()
        time_now = datetime.datetime.now()
        now_str = time_now.strftime("%Y-%m-%d %H:%M:%S")
        records = self.read_record()
        found = None

        for record in records:
            if record["Name"] == get_name and record["Time-out"] == "":
                found = record
                break

        if found is not None:
            time_in = datetime.datetime.strptime(found["Time-in"], "%Y-%m-%d %H:%M:%S")
            duration = time_now - time_in
            hours = duration.seconds // 3600
            minutes = (duration.seconds % 3600) // 60

            found["Time-out"] = now_str
            found["Duration"] = f"{hours}h {minutes}m"

            print(f"Time-out recorded at {now_str} | Duration: {found['Duration']}\n")
            time.sleep(2)
        else:
            new_record = {"Name": get_name, "Time-in": now_str, "Time-out": "", "Duration": ""}
            records.append(new_record)
            print("Time-in recorded at", now_str + '\n')
            time.sleep(2)

        self.write_record(records)

    def list_record(self):
        records = self.read_record()
        print("Listing Records...")
        time.sleep(1)

        print("---------" * 6)
        print("Name | Time-in | Time-out | Duration")
        for record in records:
            print(f"{record['Name']}: {record['Time-in']} - {record['Time-out']} |
{record['Duration']}")
        print("---------" * 6)
        time.sleep(2)

    def edit_record(self):
        get_name = input("Enter your name for editing: ").strip().title()
        records = self.read_record()
```

AGUIRRE, PAUL VINCENT S.
M001-CP102-MW-4:00-06:30

```
        user_record = [record for record in records if record['Name'] == get_name]

        if not user_record:
            print(f"No records for {get_name}\n")
            time.sleep(2)
            return

        for i, r in enumerate(user_record, 1):
            print(f"{i}: {r}")

        record_number = int(input("Enter a number to edit: "))
        record = user_record[record_number - 1]
        index = records.index(record)

        print("Select field to edit: 1-Name, 2-Time-in, 3-Time-out")
        field_choice = int(input("Enter a field to edit: "))
        fields = ["Name", "Time-in", "Time-out"]
        field = fields[field_choice - 1]
        old_value = records[index][field]

        new_value = input(f"Enter a new value for {field} (current: {old_value}):
").strip().title() or old_value
        records[index][field] = new_value

        if field in ["Time-in", "Time-out"]:
            t_in = records[index]["Time-in"]
            t_out = records[index]["Time-out"]

            if t_in and t_out:
                t_in_dt = datetime.datetime.strptime(t_in, "%Y-%m-%d %H:%M:%S")
                t_out_dt = datetime.datetime.strptime(t_out, "%Y-%m-%d %H:%M:%S")
                duration = t_out_dt - t_in_dt
                hours = duration.seconds // 3600
                minutes = (duration.seconds % 3600) // 60
                records[index]["Duration"] = f"{hours}h {minutes}m"

        self.write_record(records)
        print("Record Updated Successfully!\n")
        time.sleep(2)

    def delete_record(self):
        get_name = input("Enter the name for deletion: ").strip().title()
        records = self.read_record()
        user_record = [record for record in records if record['Name'] == get_name]

        if not user_record:
            print(f"No records for {get_name}\n")
            time.sleep(2)
            return

        for i, r in enumerate(user_record, 1):
            print(f"{i}: {r}")

        print("---------" * 6)
        record_number = int(input("Enter the record number for deletion: "))
        record = user_record[record_number - 1]

        records.remove(record)
        self.write_record(records)
        print("Record deleted successfully!")
        print("---------" * 6 + "\n")
        time.sleep(2)

    def main(self):
        while True:
            print("Welcome to Employee Clock System (Clock in and out)!")
            print("\n\t1. Clock-In/Clock-Out")
            print("\t2. Edit Timestamp")
            print("\t3. List all Timestamp Records")
            print("\t4. Delete A Timestamp Record")
            print("\t5. Exit")
```

AGUIRRE, PAUL VINCENT S.
M001-CP102-MW-4:00-06:30

```
            choice = int(input("\nEnter your choice(1-5): ").strip())

            if choice == 1:
                self.add_record()
            elif choice == 2:
                self.edit_record()
            elif choice == 3:
                self.list_record()
            elif choice == 4:
                self.delete_record()
            elif choice == 5:
                print("Exiting...")
                time.sleep(2)
                print("Thank you for using this system!")
                break
            else:
                print("Invalid choice!")
                time.sleep(1)

if __name__ == "__main__":
    AttendanceSystem().main()
```

EXAMPLE OUTPUT:

```
Creating file...
File has been created.

Welcome to Employee Clock System (Clock in and out)!

    1. Clock-In/Clock-Out
    2. Edit Timestamp
    3. List all Timestamp Records
    4. Delete A Timestamp Record
    5. Exit

Enter your choice(1-5):
```

```
Enter your choice(1-5): 1
Enter name: Vincent
Time-in recorded at 2025-03-16 20:09:05

Welcome to Employee Clock System (Clock in and out)!

    1. Clock-In/Clock-Out
    2. Edit Timestamp
    3. List all Timestamp Records
    4. Delete A Timestamp Record
    5. Exit

Enter your choice(1-5): 1
Enter name: Vincent
Time-out recorded at 2025-03-16 20:09:19 | Duration: 0h 0m

Welcome to Employee Clock System (Clock in and out)!

    1. Clock-In/Clock-Out
    2. Edit Timestamp
    3. List all Timestamp Records
    4. Delete A Timestamp Record
    5. Exit

Enter your choice(1-5):
```

```
Welcome to Employee Clock System (Clock in and out)!

    1. Clock-In/Clock-Out
    2. Edit Timestamp
    3. List all Timestamp Records
    4. Delete A Timestamp Record
    5. Exit

Enter your choice(1-5): 2
Enter your name for editing: Vincent
1: {'Name': 'Vincent', 'Time-in': '2025-03-16 20:09:05', 'Time-out': '2025-03-16 20:09:19', 'Duration': '0h 0m'}
Enter a number to edit: 1
Select field to edit: 1-Name, 2-Time-in, 3-Time-out
Enter a field to edit: 1
Enter a new value for Name (current: Vincent): Viktor
Record Updated Successfully!
```

AGUIRRE, PAUL VINCENT S.
M001-CP102-MW-4:00-06:30

```
Enter your choice(1-5): 3
Listing Records...
-----------------------------------------------------
Name | Time-in | Time-out | Duration
Viktor: 2025-03-16 20:09:05 - 2025-03-16 20:09:19 | 0h 0m
-----------------------------------------------------
Welcome to Employee Clock System (Clock in and out)!

    1. Clock-In/Clock-Out
    2. Edit Timestamp
    3. List all Timestamp Records
    4. Delete A Timestamp Record
    5. Exit

Enter your choice(1-5):
```

```
Welcome to Employee Clock System (Clock in and out)!

    1. Clock-In/Clock-Out
    2. Edit Timestamp
    3. List all Timestamp Records
    4. Delete A Timestamp Record
    5. Exit

Enter your choice(1-5): 4
Enter the name for deletion: Vincent
No records for Vincent

Welcome to Employee Clock System (Clock in and out)!

    1. Clock-In/Clock-Out
    2. Edit Timestamp
    3. List all Timestamp Records
    4. Delete A Timestamp Record
    5. Exit

Enter your choice(1-5): 4
Enter the name for deletion: Viktor
1: {'Name': 'Viktor', 'Time-in': '2025-03-16 20:09:05', 'Time-out': '2025-03-16 20:09:19', 'Duration': '0h 0m'}
-----------------------------------------------------
Enter the record number for deletion: 1
Record deleted successfully!
```

AGUIRRE, PAUL VINCENT S.
M001-CP102-MW-4:00-06:30

```
------------------------------------------------------


Welcome to Employee Clock System (Clock in and out)!

    1. Clock-In/Clock-Out
    2. Edit Timestamp
    3. List all Timestamp Records
    4. Delete A Timestamp Record
    5. Exit

Enter your choice(1-5): 5
Exiting...
Thank you for using this system!

Process finished with exit code 0
```