Application to input and store employee ratings

Description: This script provides a simple console-based employee data management system, allowing users to view, input, and save employee information, with error handling for robust operation. The application collects, processes, and saves employee data. This includes options for displaying, inputting, and saving employee information to a file.

Summary of Menu Options

1. **Display Current Data**: Shows the current employee data.
2. **Get New Data**: Allows input of new employee data and displays the updated list.
3. **Save Data**: Saves the current employee data to the file `EmployeeRatings.json`.
4. **Exit**: Ends the program.

Imports

The script imports modules that are responsible for handling data, processing, and user presentation:

- `data_classes as dat`: Contains the data models and constants for the employee data and menu options.
- `processing_classes as proc`: Handles the reading and writing of employee data from and to a file.
- `presentation_classes as pres`: Responsible for interacting with the user, including displaying menus and employee data, and collecting user input.

Constants

- **FILE_NAME** (str): Defines the filename `EmployeeRatings.json` where employee data will be stored and loaded from.
- **menu_choice** (str): Variable to store the user's choice from the displayed menu.

Main Script Logic

1. Read Employee Data from File

The script begins by attempting to load existing employee data from the file `EmployeeRatings.json`.

This calls the `read_employee_data_from_file` function from the `FileProcessor` class, which reads employee data and stores it in the `employees` variable.

---

2. Display Main Menu and Handle User Input

The main logic is enclosed in an infinite `while` loop, which continuously displays a menu to the user and processes their input until the user chooses to exit.

- **Display Menu**: The `output_menu` function from `presentation_classes` displays the available options from `dat.MENU`.
- **Input Menu Choice**: The script waits for user input, storing the choice in the `menu_choice` variable.

---

3. Menu Option 1: Display Current Data

If the user selects option "1", the script attempts to display the current employee data.

- **Display Data**: The `output_employee_data` function from `presentation_classes` is called to show the employee information.
- **Error Handling**: If there is an issue, an error message is shown using `output_error_messages`.

---

4. Menu Option 2: Input New Data and Display the Change

If the user selects option "2", the script prompts for new employee data and updates the displayed information.

- **Input New Data**: The `input_employee_data` function collects new employee data from the user and updates the `employees` variable.
- **Display Updated Data**: The `output_employee_data` function is called again to display the updated employee list.

---

5. Menu Option 3: Save Data to File

If the user selects option "3", the script saves the employee data back into the file `EmployeeRatings.json`.

- **Save Data**: The `write_employee_data_to_file` function saves the employee data to the specified file.
- **Confirmation**: A message is printed to indicate that the data was saved successfully.

---

6. Menu Option 4: Exit the Program

If the user selects option "4", the script terminates the loop and ends the program.

- **Exit Program**: The `break` statement terminates the `while` loop, effectively ending the program.

Error Handling

Throughout the script, the following error-handling mechanism is used:

- **Try-Except Blocks**: Most actions (reading, writing, displaying, and inputting data) are wrapped in try-except blocks to handle potential exceptions. If an exception occurs, an error message is displayed using the `output_error_messages` function from `presentation_classes`.
