Adam Eichholz
September 10, 2025
Foundations of Programming: Python
Assignment 8

Employee Ratings README

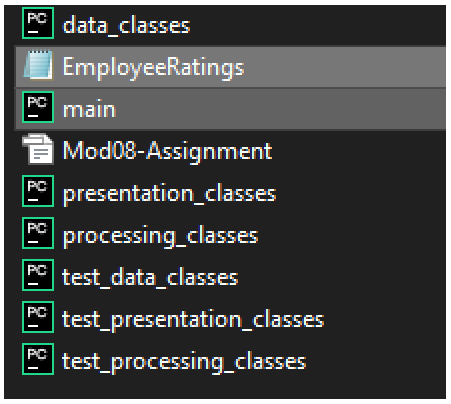
Introduction

The Employee Ratings application is a tool to compile and display Employee Rating data. Data saved includes first and last name, date of review, and review rating on a scale of 1 through 5. Data can be saved so that it can be retrieved for future use.

Main Application

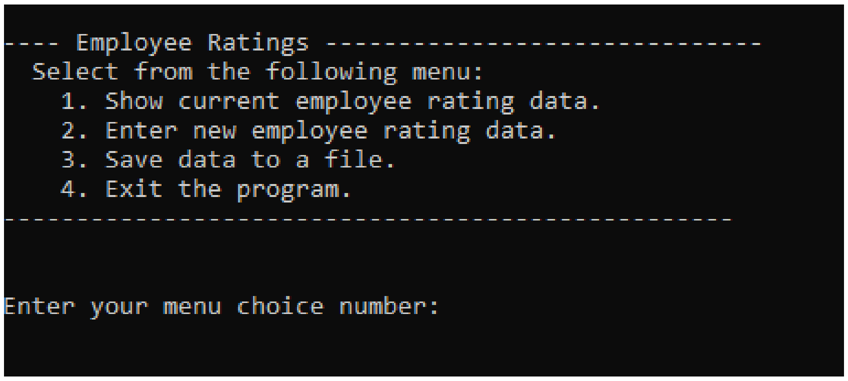
Startup

To launch the Employee Ratings application, use the *main* file. The program should automatically load saved data from the file *EmployeeRatings.json*.



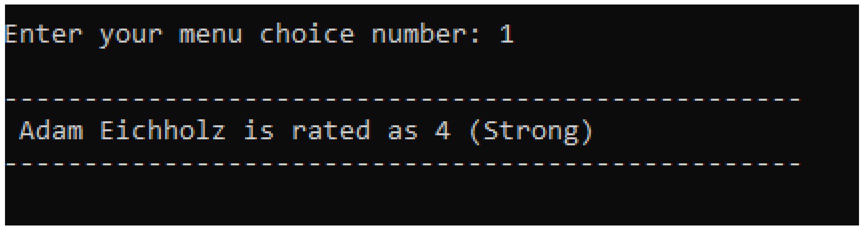
Main Menu

The program's main menu presents four options:



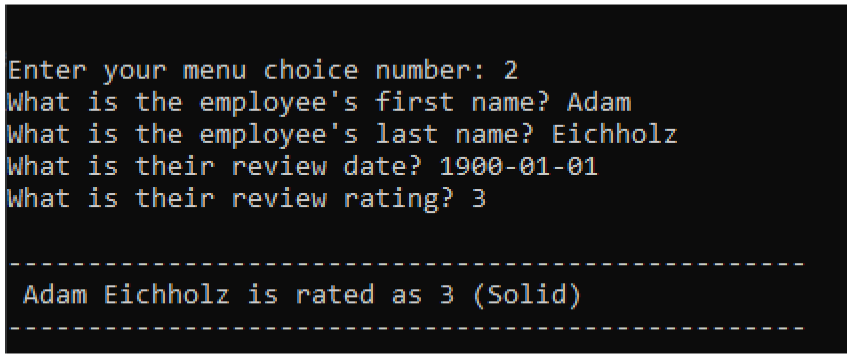
```
---- Employee Ratings ----------------------------
  Select from the following menu:
    1. Show current employee rating data.
    2. Enter new employee rating data.
    3. Save data to a file.
    4. Exit the program.
--------------------------------------------------


Enter your menu choice number:
```

1. **Show current employee rating data.**
    - This option will display information that has been saved in the *EmployeeRatings.json* file.
    - Information displayed will include first and last name, and the review rating. Note that the review date is saved, but is not displayed.



```
Enter your menu choice number: 1

--------------------------------------------------
 Adam Eichholz is rated as 4 (Strong)
--------------------------------------------------
```

2. **Enter new employee rating data.**
    - Enters a new record that is held locally. Note that the data has not yet been saved!
    - First and last name, review date, and review rating are entered. The entered information will then be displayed.



```
Enter your menu choice number: 2
What is the employee's first name? Adam
What is the employee's last name? Eichholz
What is their review date? 1900-01-01
What is their review rating? 3

--------------------------------------------------
 Adam Eichholz is rated as 3 (Solid)
--------------------------------------------------
```

3. **Save data to a file.**
    - Saves any locally saved data to the *EmployeeRatings.json* file.
    - Data is appepended; previously entered data will not be replaced.

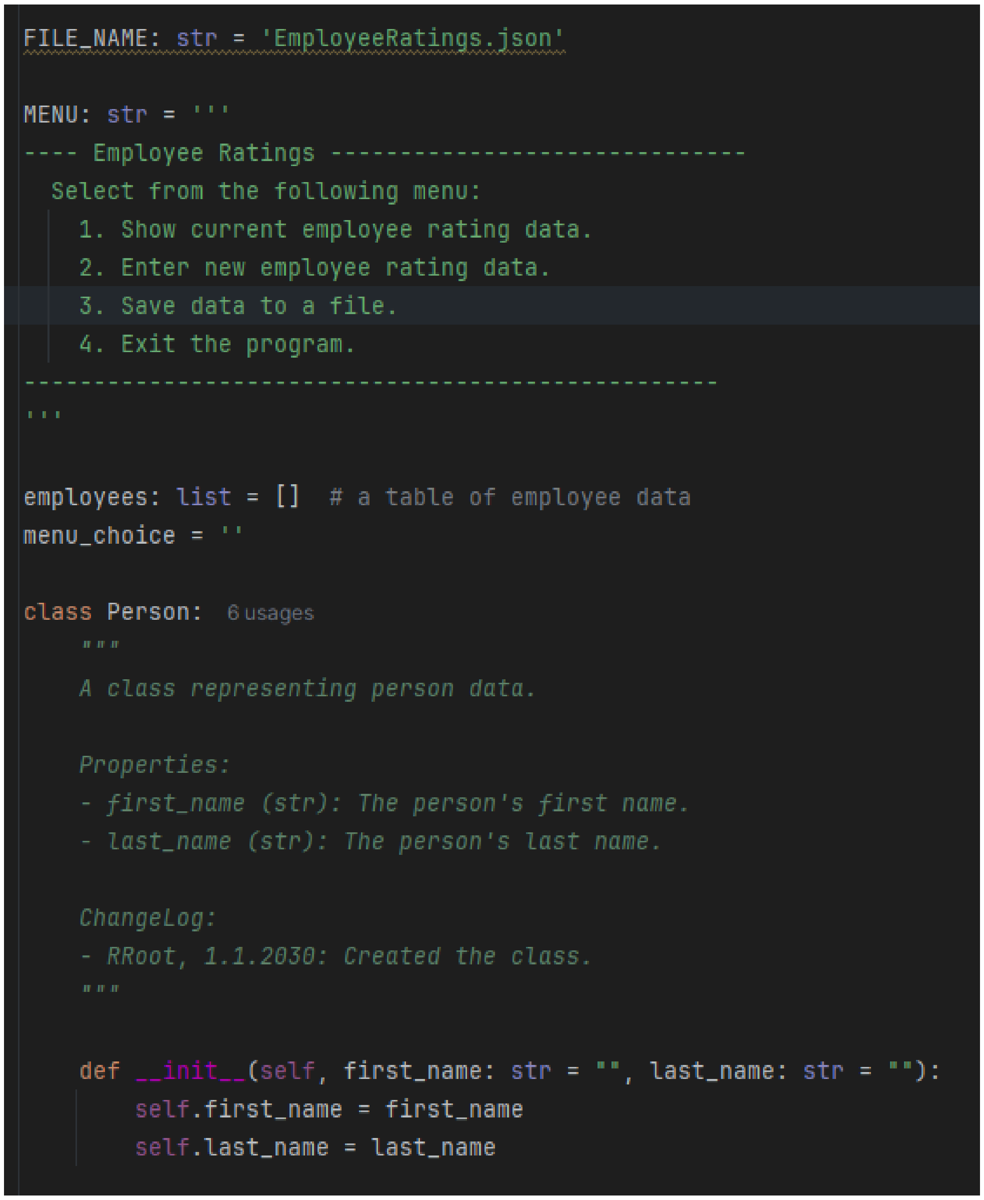
4. **Exit the program.**
    - Note that any data that has not yet been saved using Option 3 will be deleted.

Classes

Application functions have been organized into three classes: *Data, Presentation,* and *Processing.*

Data

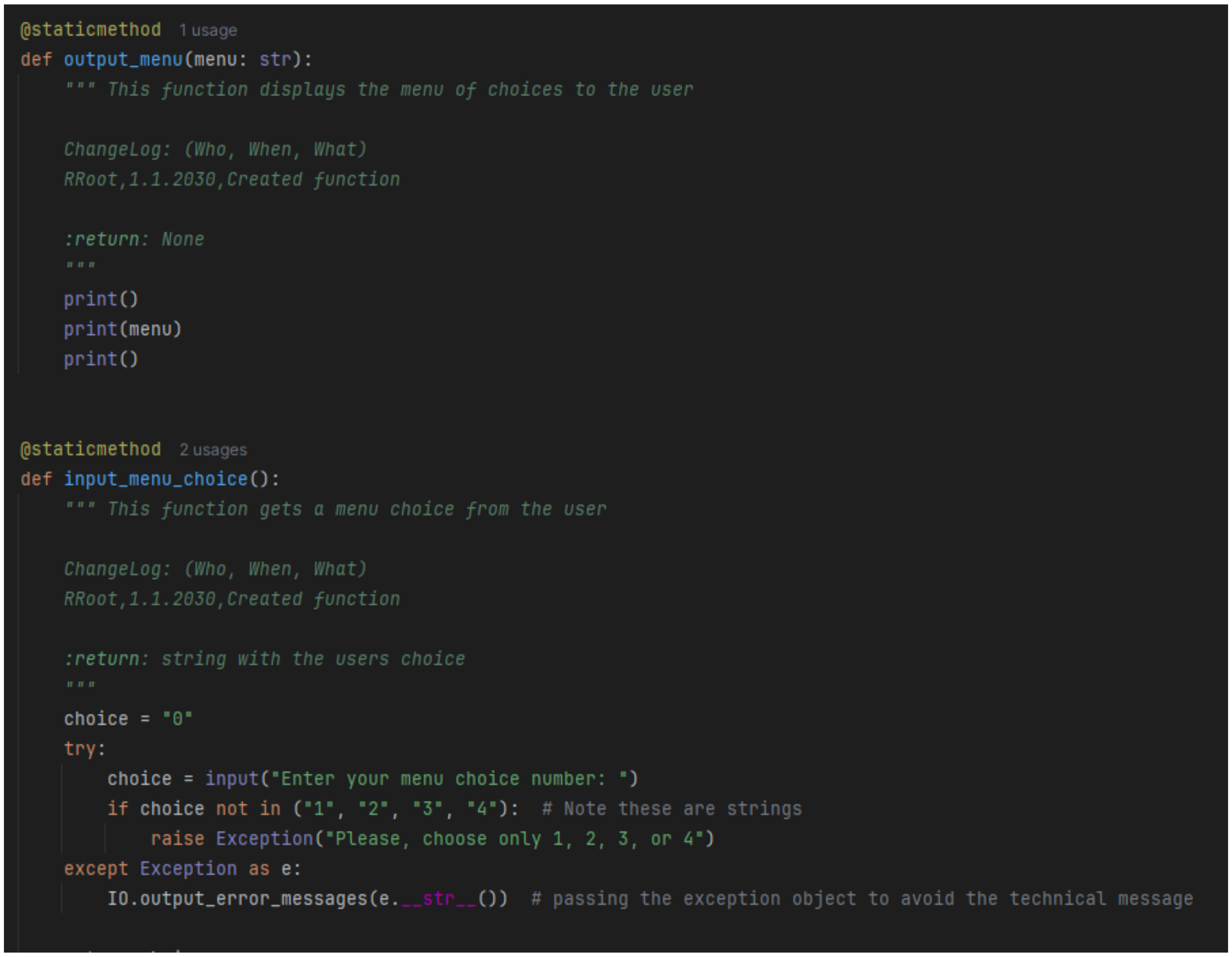
This layer holds the classes to import saved data (the *EmployeeRatings.json* file), as well as code classes for the *Person* and *Employee* objects. Any updates or changes to how data is organized should be placed into this layer.



```
FILE_NAME: str = 'EmployeeRatings.json'

MENU: str = '''
---- Employee Ratings ------------------------------
  Select from the following menu:
    1. Show current employee rating data.
    2. Enter new employee rating data.
    3. Save data to a file.
    4. Exit the program.
--------------------------------------------------
'''

employees: list = []  # a table of employee data
menu_choice = ''

class Person:  6 usages
    """
    A class representing person data.

    Properties:
    - first_name (str): The person's first name.
    - last_name (str): The person's last name.

    ChangeLog:
    - RRoot, 1.1.2030: Created the class.
    """

    def __init__(self, first_name: str = "", last_name: str = ""):
        self.first_name = first_name
        self.last_name = last_name
```

Presentation

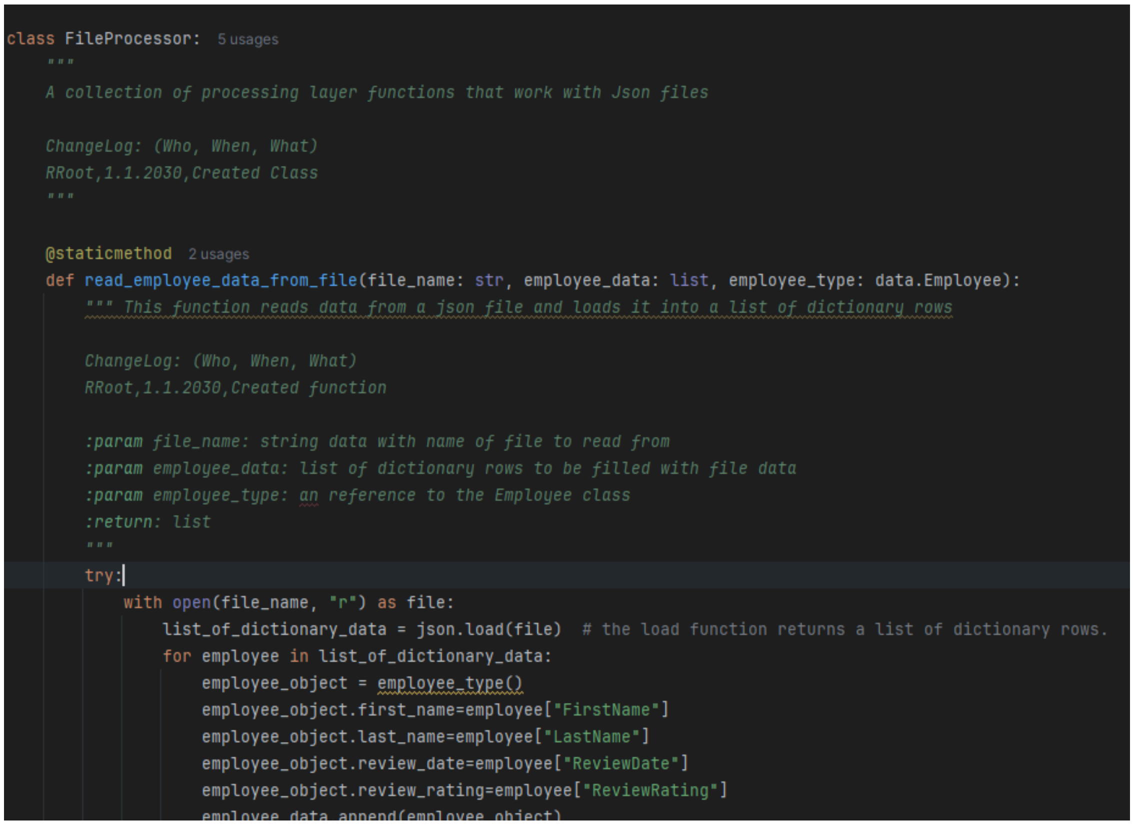
This layer holds the classes used for the input/output functions of the application. These classes display the main menu, option selections, previously saved data, and newly entered data. This layer also includes exception handling for any errors that might be triggered while running the program. Any changes to the input/output functions of the program should be made here.



```
@staticmethod  1 usage
def output_menu(menu: str):
    """ This function displays the menu of choices to the user

    ChangeLog: (Who, When, What)
    RRoot,1.1.2030,Created function

    :return: None
    """
    print()
    print(menu)
    print()


@staticmethod  2 usages
def input_menu_choice():
    """ This function gets a menu choice from the user

    ChangeLog: (Who, When, What)
    RRoot,1.1.2030,Created function

    :return: string with the users choice
    """
    choice = "0"
    try:
        choice = input("Enter your menu choice number: ")
        if choice not in ("1", "2", "3", "4"):  # Note these are strings
            raise Exception("Please, choose only 1, 2, 3, or 4")
    except Exception as e:
        IO.output_error_messages(e.__str__())  # passing the exception object to avoid the technical message
```

Processing

This layer holds the classes used for the applications' command functions. These include the FileProcessor class, which reads and writes data while using the application. Any changes to how the application should process data should be made here.



```python
class FileProcessor:  5 usages
    """
    A collection of processing layer functions that work with Json files

    ChangeLog: (Who, When, What)
    RRoot,1.1.2030,Created Class
    """

    @staticmethod  2 usages
    def read_employee_data_from_file(file_name: str, employee_data: list, employee_type: data.Employee):
        """ This function reads data from a json file and loads it into a list of dictionary rows

        ChangeLog: (Who, When, What)
        RRoot,1.1.2030,Created function

        :param file_name: string data with name of file to read from
        :param employee_data: list of dictionary rows to be filled with file data
        :param employee_type: an reference to the Employee class
        :return: list
        """
        try:
            with open(file_name, "r") as file:
                list_of_dictionary_data = json.load(file)  # the load function returns a list of dictionary rows.
                for employee in list_of_dictionary_data:
                    employee_object = employee_type()
                    employee_object.first_name=employee["FirstName"]
                    employee_object.last_name=employee["LastName"]
                    employee_object.review_date=employee["ReviewDate"]
                    employee_object.review_rating=employee["ReviewRating"]
                    employee_data.append(employee_object)
```

Testing Classes

This application also includes three testing environments, one for each layer: testing for data classes, presentation classes, and processing classes. These layers can be used to test any changes made to the class layers before running the primary application.