Assignment 3
Subscription Manager

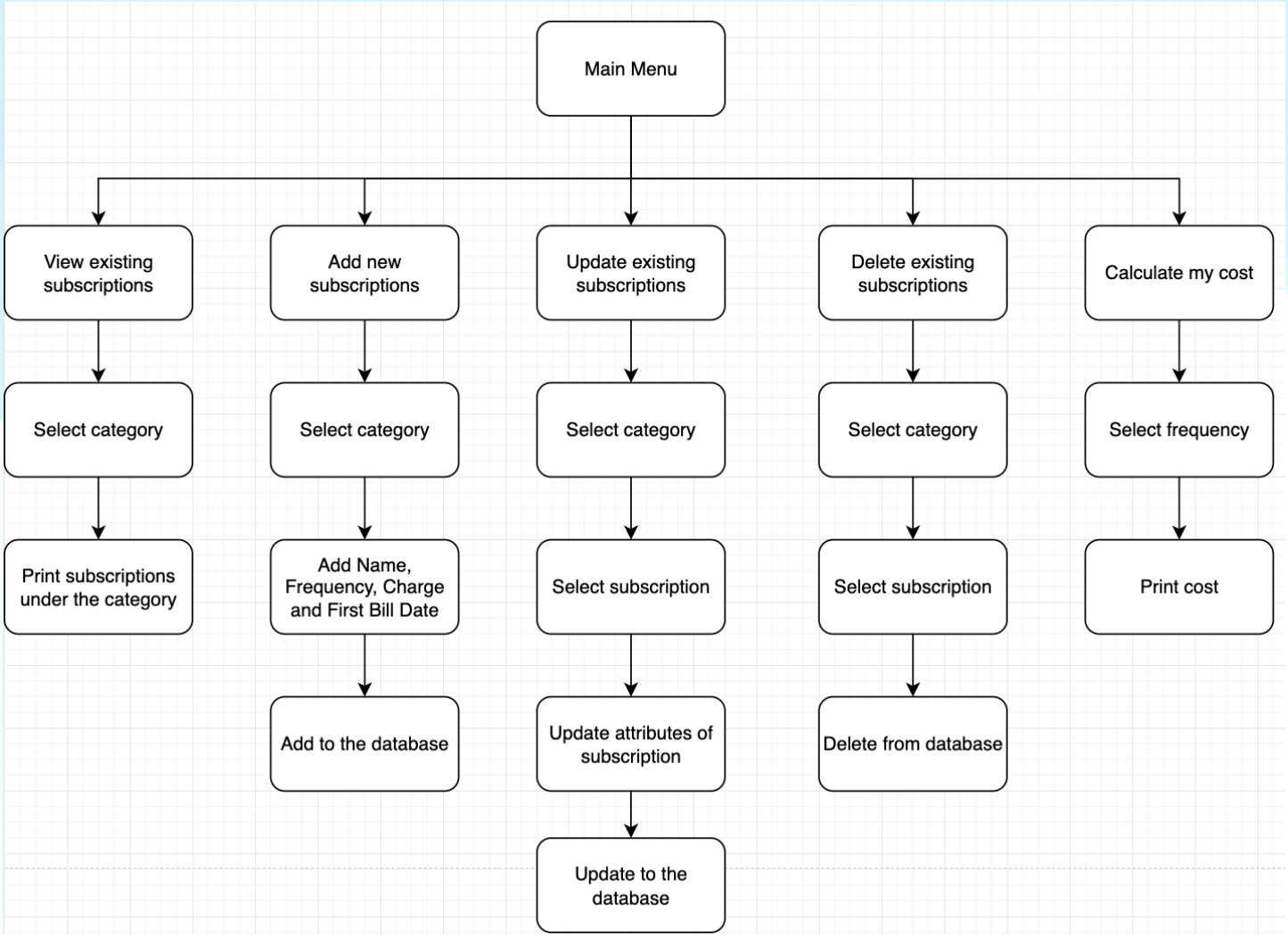
Ji Li 17/12/2022

Table of Content

Code Main Structure

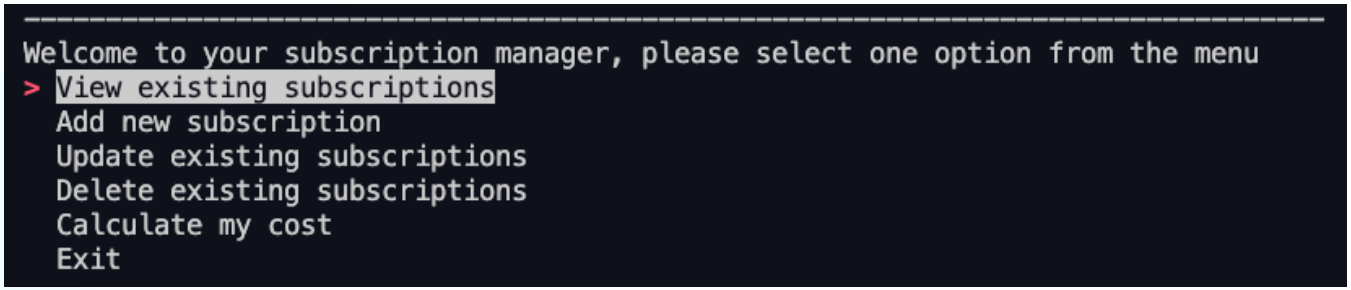
- *data/subscription.json*: database to store existing subscriptions
- *project/json_handling.py*: includes CRUD functions for JSON files
- *project/terminal_menu.py*: a function to show a menu in the terminal, with given options and prompt messages
- *project/subscription_handling.py*: defines a Subscription class, which contains all CRUD methods for subscriptions
- *project/main.py*: a Python script to connect all the scripts described above
- *./run_application.sh*: a bash script to activate venv, install required packages, check Python version and run main.py

Application Main Structure



How to Run the Application

- Make sure Python 3 is installed
- Make sure the current directory is in *src/*
- Run *chmod +x run_application.sh* to add execute permission to this file
- *Run ./run_application.sh* in the terminal
- Main menu will be printed in the terminal and the application is ready to use



Feature - Terminal Menu

- Terminal Menu is applied throughout the whole project, which makes the application easier to use, and can reduce errors caused by human input (package: simple-term-menu)

```
Welcome to your subscription manager, please select one option from the menu
> View existing subscriptions
  Add new subscription
  Update existing subscriptions
  Delete existing subscriptions
  Calculate my cost
  Exit
```

- Error handling

```python
def terminal_menu(options, prompt):
    print(prompt)
    while True:
        try:
            terminal_menu = TerminalMenu(options)
            menu_entry_index = terminal_menu.show()
            selected = options[menu_entry_index]
            print(selected)
            return selected
            break
        # error handling when user inputs invalid value, instead of selecting one
        except TypeError:
            print('Please select one option from of the menu')
```

Feature - View Existing Subscriptions

- Users can select ‘View All’ to check all existing subscriptions
- Alternatively, users can select an existing category to view all subscriptions under this category

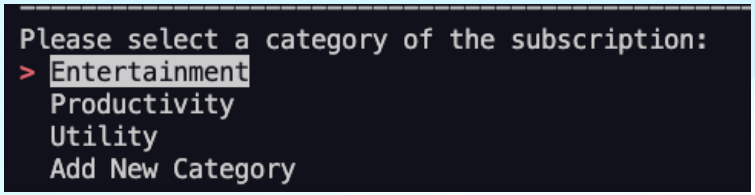
```
--------------------------------------------------------------------------------
Please select a category of the subscription:
> View All
  Entertainment
```

- Subscriptions will be pretty-printed in the terminal (package: json)

```
--------------------------------------------------------------------------------
Please select a category of the subscription:
Entertainment
[
    {
        "Name": "Spotify",
        "Frequency": "Quarterly",
        "Charge": 12.0,
        "First bill date": "2022-12-12"
    },
    {
        "Name": "Netflix1",
        "Frequency": "Monthly",
        "Charge": 12.0,
        "First bill date": "2020-12-12"
    }
]
```

Feature - Add New Subscription

- Users can add a new subscription under existing categories or add a new category
- A default category list (Entertainment, Productivity and Utility) is also available to select



- No duplicated categories are allowed (spaces at the beginning and end will be trimmed and case-insensitive)

```python
# if user would like to add new category, ask for new category name
if category_selected == 'Add New Category':
    category_selected = input(
        'Please enter the name of the new category: ').strip()
    # data validation, when input is empty
    while category_selected == '':
        category_selected = input(
            'Please enter a valid category name: ').strip()

    # avoid duplicates
    for category in category_option:
        if category_selected.lower() == category.lower():
            category_selected = category
            break
```

Feature - Add New Subscription

- Name: no duplicated names are allowed, user will be asked to input another name

```
while True:
    # data validation, when input is empty
    if name == '':
        name = input('Please enter a valid name: ')
    # avoid duplicates
    elif name.lower().strip() in [sub.lower() for sub in existing_subscription]:
        name = input(
            'Subscription exists! Please enter a different name: ').strip()
    else:
        break
```

- Frequency: can only be selected from the given options.

```
Please select the frequency:
> Daily
  Monthly
  Quarterly
  Annual
```

Feature - Add New Subscription

- Charge: must be a positive numerical value

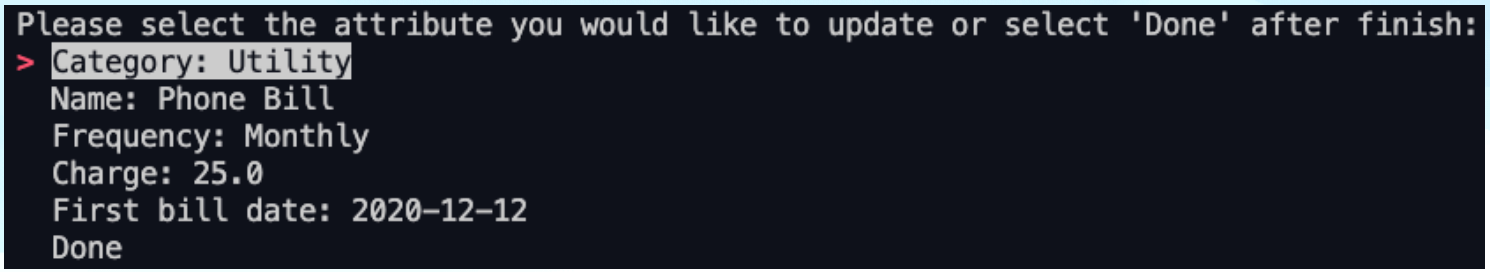
```
while True:
    try:
        charge = float(
            input('Please enter the charge of the subscription: '))
        if charge < 0:
            print('Please enter a positive number!')
            continue
    # if user doesn't enter a valid number, ask again
    except ValueError:
        print('Please enter a valid number!')
        continue
    else:
        break
```

- First bill date: must be in a date format of yyyy-mm-dd (package: datetime)

```
while True:
    # error handling, when user didn't input a valid date
    try:
        first_bill_date = datetime.strptime(
            user_input, '%Y-%m-%d').date()
        break
    except ValueError:
        user_input = input(
            'Please enter a valid date in the format of yyyy-mm-dd: ')
        continue
    else:
        break
```

Feature - Update Existing Subscriptions

- Users can keep updating any of the attributes of an existing subscription. Feature realised by a while loop



- DRY principle: All data validations are the same with ‘Add New subscriptions’, which avoid duplicates of codes

Feature - Delete Existing Subscriptions

- Confirmation before deleting the subscription

```
Are you sure to delete {'Name': 'Phone Bill', 'Frequency': 'Monthly', 'Charge': 25.
0, 'First bill date': '2020-12-12'}?
> Yes
  No
```

- Category will be deleted as well when there is no active subscriptions under it

```python
def delete_json(category, subscription, filepath):
    file_data = read_json(filepath)
    file_data[category].remove(subscription)

    # if no subscription exists under the category, delete the category
    if file_data[category] == []:
        del file_data[category]

    with open(filepath, 'w') as file:
        json.dump(file_data, file)
```

Feature - Calculate my cost

- Users to select the frequency
- Terminal prints out the calculation result

```
--------------------------------------------------------------------------------
Please select the frequency:
Monthly
Based on your subscriptions, your estimated monthly cost is $41.0
```

Feature - Return main menu

- After finishing using one of the features, the user can press ENTER to return to main menu or press anything else to exit the program. (package: getkey)

```
--------------------------------------------------------------------------------
Press ENTER to go back to the main menu, or press anything else to exit the application
```

Development Process Review

- DRY principle
  - Make sure not repeat yourself
  - Help clean and re-organise the codes

- Testing and error handling
  - Start testing the codes ASAP
  - Try to cover all possible errors

Development Process Review

- My favourite part: terminal menu
  - Interactive with the user
  - Reduce human input errors

- Limitations and future expectations
  - Add username and password login feature
  - Allow user to reverse previous actions

Thank you!

Ji Li 17/12/2022