1. **Personal information**

Name: Dat Le

Student number: 837639

Study program: Master's Programme in Automation and Electrical Engineering

Study year: 2022

Date: 25/2/2022

2. **General description**

In this course, I will build an application helps managing money by capturing and grouping the transaction's data from the recorded file of different banks.

The program will support at least recorded files from following banks:

- OP
- Danske Bank
- Nordea
- S-pankki

In addition, multiple formats are also supported:

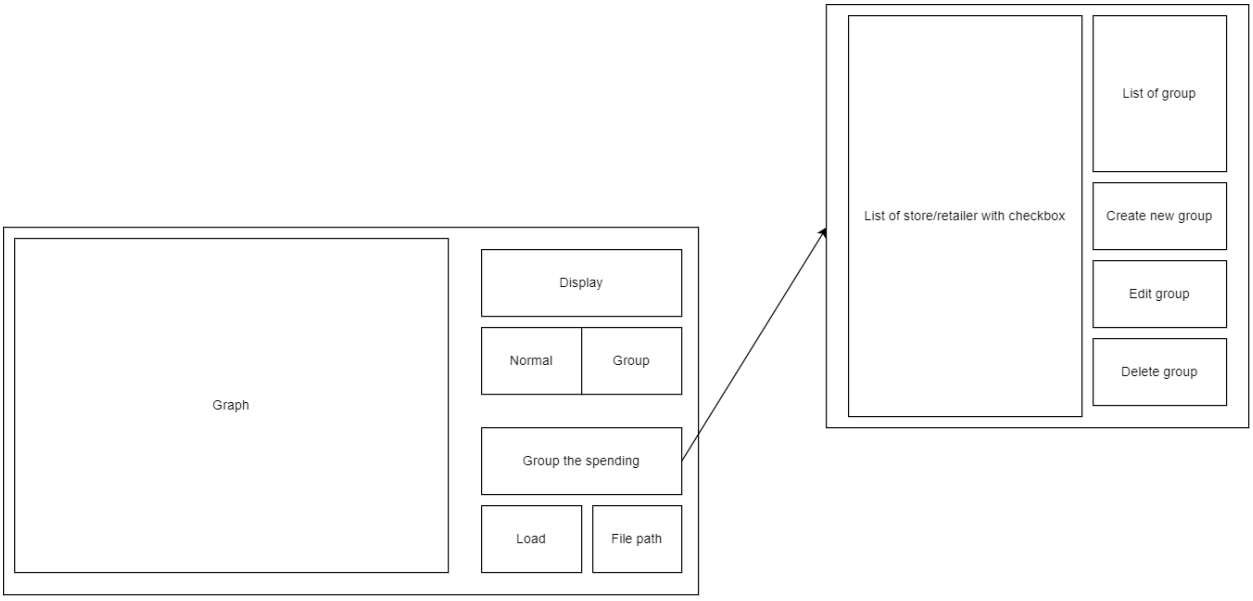
- .csv
- .xml
- .pdf: This format will be converted into .csv

The application will support following features:

- **Load data:**
    - Able to recognize the extension of file and verify whether it is supported or not
    - Able to read/convert the file to supported format (.csv and .xml)
- **Parse data**:
    - Parse given data files into individual transactions
    - Parse the individual transaction to capture needed information
        - Name of receiver
        - Amount of money
        - Date
    - Save them to defined database bases on spent date
- **Calculate spending**:
    - Calculate the spending spent at a specific store/retailer
    - If multiple store/retailers are grouped together, able to calculate the total spending
- **Grouping function**:
    - Group spending store into group with user-defined name
    - Delete the group if needed
- **Display data**: *Pie charts* are used

- Usage: Display the spending spent as individual store/retailer or user-defined group. Only **one** mode works at a time.
    - Grouping function is disabled (default): Display the spending spent at each store/retailer
    - Grouping function is enabled: Display the spending spent of different user-defined groups.
- Extra option: display based on interval
    - The graph will show the interval of available data
    - User can choose the period to display data

3. Use case description and draft of the user interface

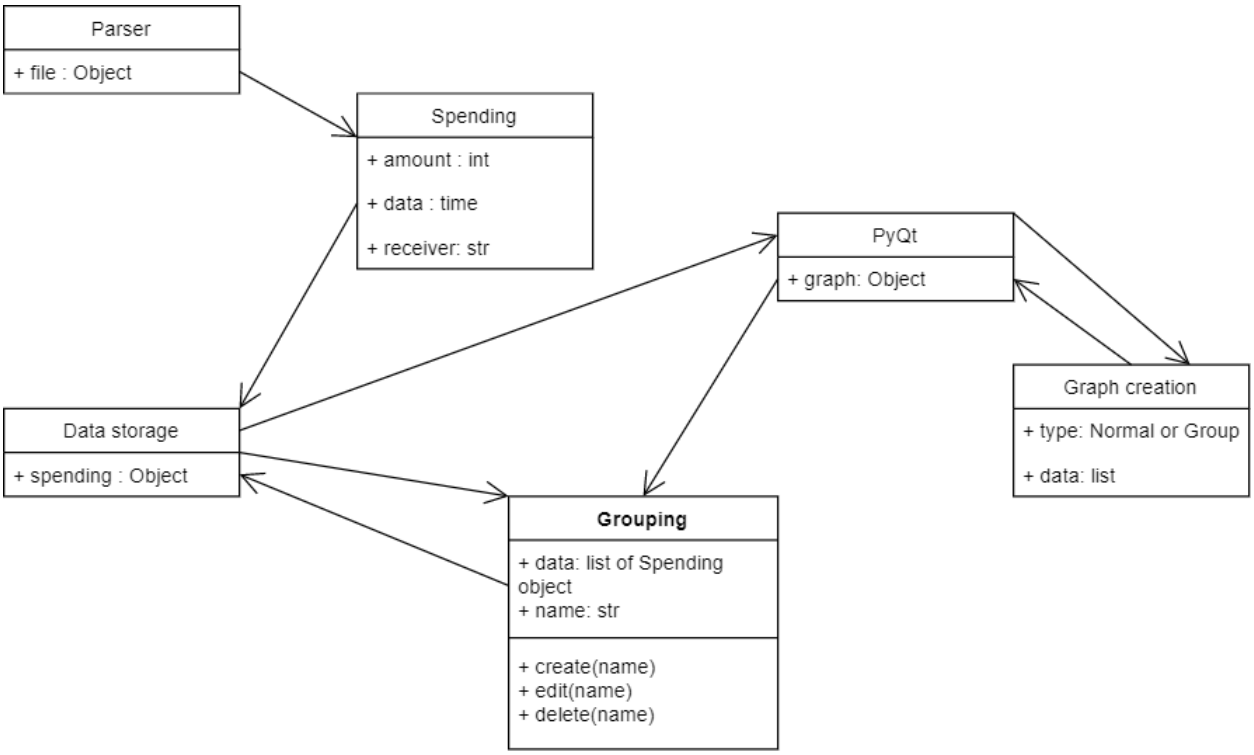


The above picture is the draft version of the application UI.

- Load data file and display:
    - User clicks the **File path** button; a window will appear and allow user to browse through file system to choose desired file.
    - User presses the **Load** button to display data. The button will have color **Red** when data is calculated and loaded, then it turns to **Green** when data is fully parsed and saved
    - User can see the visualized version of data by pressing button **Display**
    - The **Normal** and **Group** will be two checkboxes and only one can be enabled at a time.
        - **Normal** is checked: User can view spending spent at individual stores/retailers.
        - **Group** is checked: User can view the spending spent as user-defined groups.
- Grouping function:
    - This function will be disabled until there is data in “database” of the application – achieved by loading data with **File path** and **Load** buttons.

- User presses the **Group the spending** and a new window appears
- This window has function to create/edit/delete user defined group

4. Program's structure plan



5. Data structures

I think list and dictionary data types are enough for storing/reading data in this application

6. Files and file format

The application will work on files with .csv and .xml format

7. Algorithms

Since the application is used to display the spending, there aren't many algorithms needed when simple subtractions are enough for calculation.

8. Testing plan

I will try to implement the test-driven development approach in developing this application by writing the unit tests for functions before the function is fully implemented. The test scenarios will be gotten directly from the usage of this application.

About the integration testing, I will divide the testing schedule into different sprints – where small functionality of the application is finalized. Then the test in integration level is executed on relevant functions to check their cooperation.

9. **Libraries and other tools**

I didn't think about the usage of external libraries yet. I will discuss with my assistant in demo section.

10. **Schedule**

From my current point of view, the schedule will be divided into:

- Checkpoint 1: The application can load/parse data
- Checkpoint 2: The application can display data with normal option
- Checkpoint 3: The application supports the grouping function
- Checkpoint 4: The application can display the data with normal or group option
- Checkpoint 5: The application allows user to choose the interval to display the spending

11. **Literature references and links**

- Stackoverflow
- Python official documentation
- PyQt's documentation

12. **Attachments**