Financial Ghost - User Guide

By: CS2113T-W12-2   Since: September 2019 Licence: NUS

1. Introduction

Financial Ghost is a single-user application that tracks multiple financial entities and helps manage finances.
With an aging population and a falling birth rate, the risk of youths being part of what is known as a Sandwich Generation is becoming increasingly prevalent in Singapore. The best solution to any problem is prevention rather than cure. Therefore, we believe the best way to prevent one from being part of the sandwich generation is to start planning your finances as early as possible.
Financial Ghost is a financial planner targeted towards the fresh graduates of NUS. As members of Singapore's society who are about to enter the workforce and take on additional responsibilities and independence, they represent a demographic that is at risk of becoming a Sandwich Generation.
A Command Line Interface (CLI) application with a Graphical User Interface (GUI) display, Financial Ghost is optimised for individuals who can type efficiently. Coded in

Java, this desktop application can run on any machine with Java Runtime Environment installed. If you can type fast, Financial Ghost can get your finance management done faster than traditional GUI apps. Interested? Jump to the Section 2, “Quick Start” to get started!

2. Quick Start

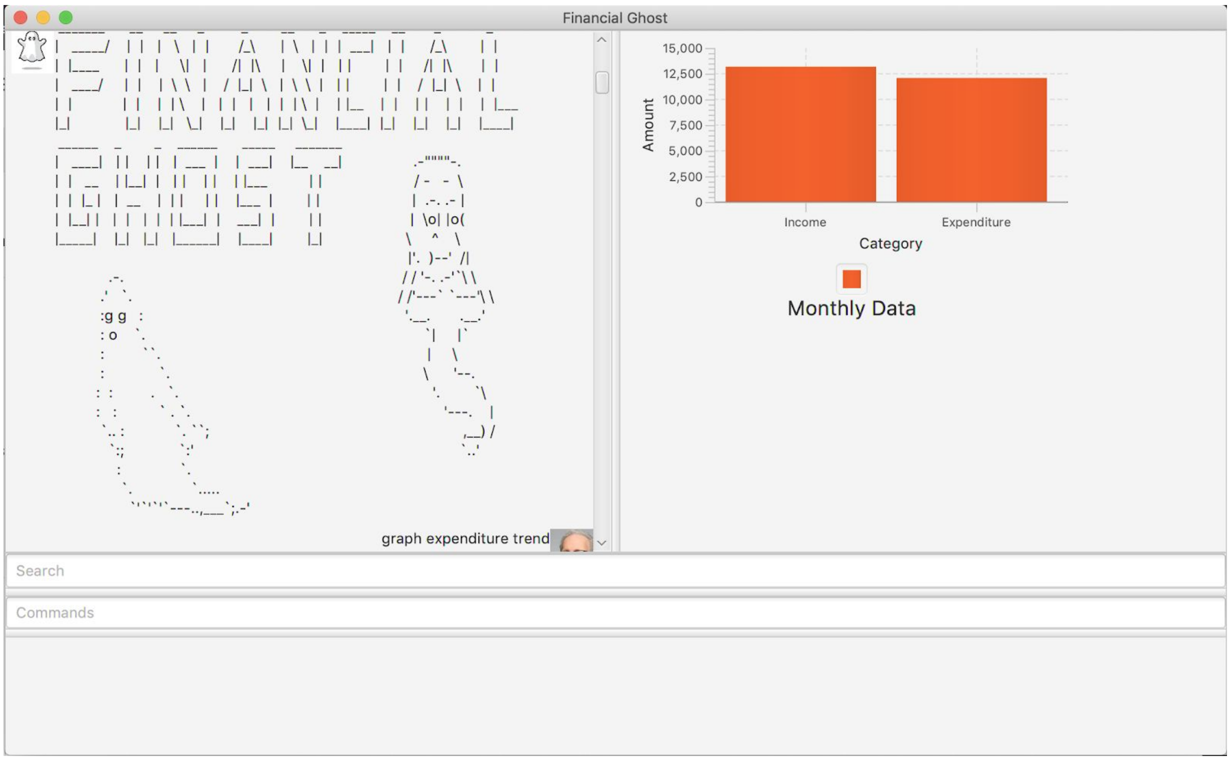
This section serves as a tutorial for you to familiarize yourself with the usage of Financial Ghost. By following the steps below, getting started with Financial Ghost should be a fast and smooth process.

2.1. Installation

1. Ensure you have Java version 11 or later installed in your computer.
2. Download the latest version of FinancialGhost.jar here: https://github.com/AY1920S1-CS2113T-W12-2/main/releases
3. Copy the file to the folder you want to use as the home folder for your Financial Ghost.
4. Double-click the file to start the app. You should see the GUI (Graphical User Interface) in a few seconds as shown below.

2.2. User Interface

The mockup of the GUI is shown in the image below.



The table below summarises the function of each component.

Command Bar	The Text Bar on the bottom with prompt text “Commands” inside is where the Commands is being typed by the user.
Command Result Panel	As seen on top left, this is the panel where the result of the command entered by the user is shown.
Search Bar	The Search Bar is right on top of the command bar with the prompt text “Search” inside the bar. This is where searching of all the finances will be done.
Account panel	This is the panel on the right. This will be where the information of the account will be shown, whether it is through searching by the user or through command by the user.

2.3. Executing Commands

Below are some steps to teach the user how to use the application.

1. Follow the on screen steps if you are a first time user.
2. Type in the command `init [existing savings] [Avg Monthly Expenditure]` (e.g `init 30000 200`), then press Enter.
3. Type the Commands (Case Sensitive) in the Command Bar and press Enter to execute it.
4. Some examples you can try:
    a. `add income TA /amt 560 /on 10/12/2040`: Record an income of $560 from a source with description TA which was received on 10/12/2040.
    b. `spent boat noodles /amt 20 /cat food /on 15/08/2019`: Record an expenditure of $20 spent on boat noodles on 15/08/2019 which is under the category of food.
5. Previous commands can be retrieved using the arrow keys.

For the details of each command, you may refer to the Features Section below.

3. Features [v1.4]

This section shows the features and its relevant commands as of version 1.4 of Financial Ghost.

3.1. Initialising the Program

This section contains the command to initialise Financial Ghost.

3.1.1 Initialise: `init`

Allows a first time user to initialise Financial Ghost by storing their existing savings and their average monthly expenditure.

Format: `init [existing savings] [Avg Monthly Expenditure]`

Examples:

- `init 30000 200`

3.2. Managing Income and Expenditure

This section contains the commands to manage income and expenditure.
The primary function of any financial planner is to track a user's monetary inflow and outflow. This feature allows users to log their income sources and expenditures within the application and display them back to the user in an organised manner according to his or her specifications.

3.2.1 Add an income source: `add income`

Allows the user to track total income for a holistic evaluation of total money inflow

Adds an income source to the list of total income sources.

Format: `add income [description] /amt [amount in dollars] /on d/m/yyyy`

Examples:

- `add income TA /amt 480 /on 1/10/2019`
- `add income Tuition /amt 600 /on 16/10/2019`
- `add income Job /amt 4000 /on 25/9/2025`

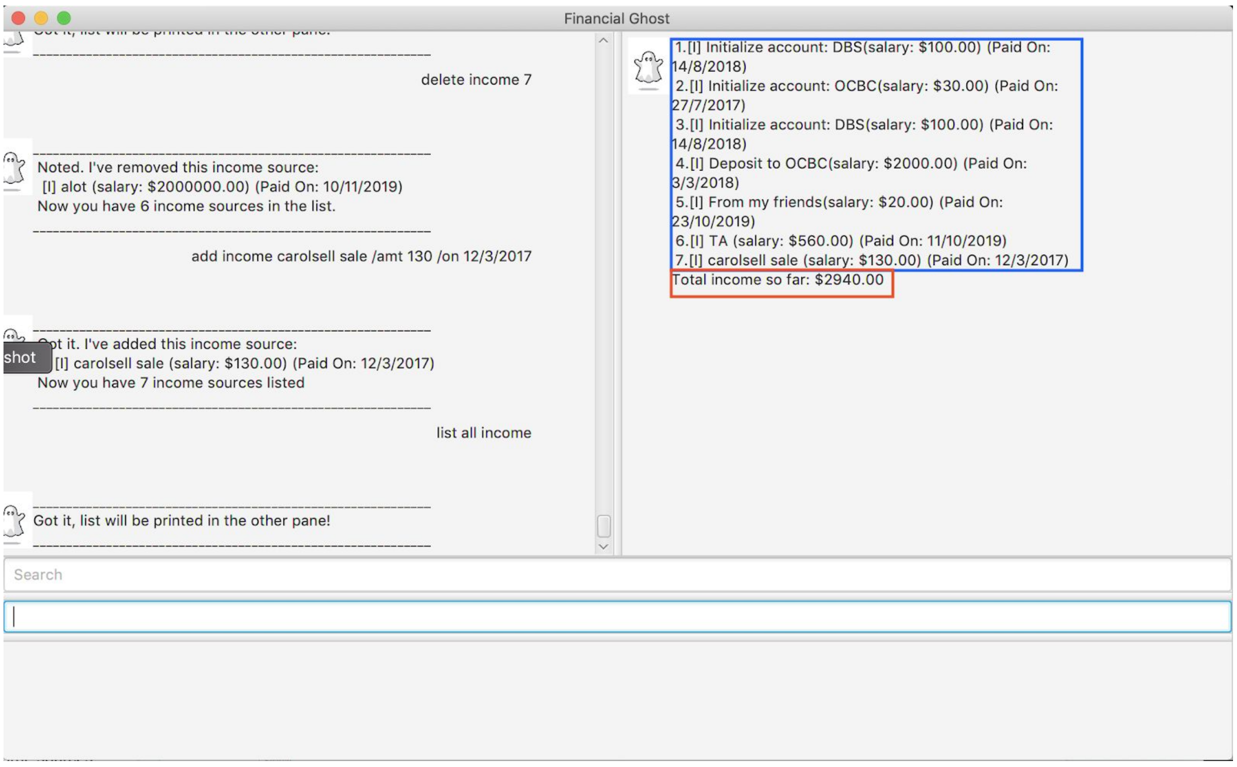
3.2.2 List all income sources: `list all income`

List all income sources added by the user according to the sequence they were added

Format: `list all income`

Example:

- User enters `list all income` into the command bar.
- The income list will be displayed in the pane on the right as enclosed by the blue box
- The total monetary inflow from all income sources is computed and displayed below the list as enclosed by the red box



```
delete income 7

Noted. I've removed this income source:
 [I] alot (salary: $2000000.00) (Paid On: 10/11/2019)
Now you have 6 income sources in the list.

add income carolsell sale /amt 130 /on 12/3/2017

Got it. I've added this income source:
 [I] carolsell sale (salary: $130.00) (Paid On: 12/3/2017)
Now you have 7 income sources listed

list all income

Got it, list will be printed in the other pane!
```

```
1.[I] Initialize account: DBS(salary: $100.00) (Paid On: 14/8/2018)
2.[I] Initialize account: OCBC(salary: $30.00) (Paid On: 27/7/2017)
3.[I] Initialize account: DBS(salary: $100.00) (Paid On: 14/8/2018)
4.[I] Deposit to OCBC(salary: $2000.00) (Paid On: 3/3/2018)
5.[I] From my friends(salary: $20.00) (Paid On: 23/10/2019)
6.[I] TA (salary: $560.00) (Paid On: 11/10/2019)
7.[I] carolsell sale (salary: $130.00) (Paid On: 12/3/2017)
Total income so far: $2940.00
```

3.2.3 Delete an income source: `delete income`

Allows the user to delete a selected income source from the total income list by index.

Format: `delete income [index]`

Examples:

- `delete income 1`

Note: The index refers to the index of the income source within the total income list.

3.2.4 Add an expenditure: `spent`

Allows users to track total expenditure for a holistic evaluation of total money outflow

Adds an expenditure to the list of total expenditures.

Format: `spent income [description] /amt [amount in dollars] /cat [category] /on d/m/yyyy`

Examples:

- `spent Menya Sakura /amt 13.50 /cat food /on 8/7/2019`
- `spent A&F shirt /amt 60 /cat apparel /on 3/3/2018`
- `spent flowers /cat present /amt 59 /on 14/2/2018`

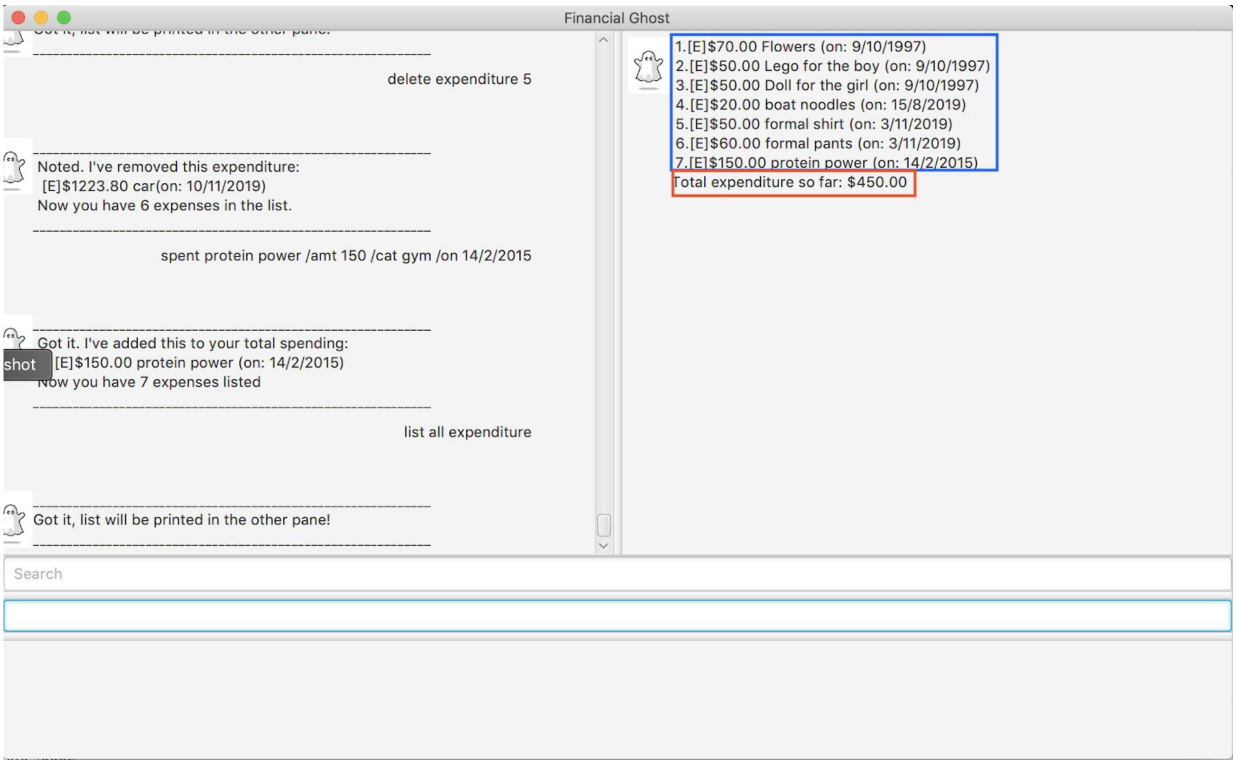
3.2.5 List all expenditures: `list all expenditure`

List all expenditures added by the user according to the sequence they were added

Format: `list all expenditure`

Example:

- User enters `list all expenditure` into the command bar.
- The expenditure list will be displayed in the pane on the right as enclosed by the blue box
- The total monetary outflow from all expenditures is computed and displayed below the list as enclosed by the red box



```
Financial Ghost

------------------------------------------------------------------
                                        delete expenditure 5

------------------------------------------------------------------
Noted. I've removed this expenditure:
 [E]$1223.80 car(on: 10/11/2019)
Now you have 6 expenses in the list.
------------------------------------------------------------------
              spent protein power /amt 150 /cat gym /on 14/2/2015

------------------------------------------------------------------
Got it. I've added this to your total spending:
 [E]$150.00 protein power (on: 14/2/2015)
Now you have 7 expenses listed
------------------------------------------------------------------
                                           list all expenditure

------------------------------------------------------------------
Got it, list will be printed in the other pane!
------------------------------------------------------------------
```

```
1.[E]$70.00 Flowers (on: 9/10/1997)
2.[E]$50.00 Lego for the boy (on: 9/10/1997)
3.[E]$50.00 Doll for the girl (on: 9/10/1997)
4.[E]$20.00 boat noodles (on: 15/8/2019)
5.[E]$50.00 formal shirt (on: 3/11/2019)
6.[E]$60.00 formal pants (on: 3/11/2019)
7.[E]$150.00 protein power (on: 14/2/2015)
Total expenditure so far: $450.00
```

Search

3.2.6 Delete an expenditure: `delete expenditure`

Allows the user to delete a selected expenditure from the total expenditure list by index.

Format: `delete expenditure [index]`

Examples:

- `delete expenditure 1`

Note: The index refers to the index of the expenditure within the total expenditure list.

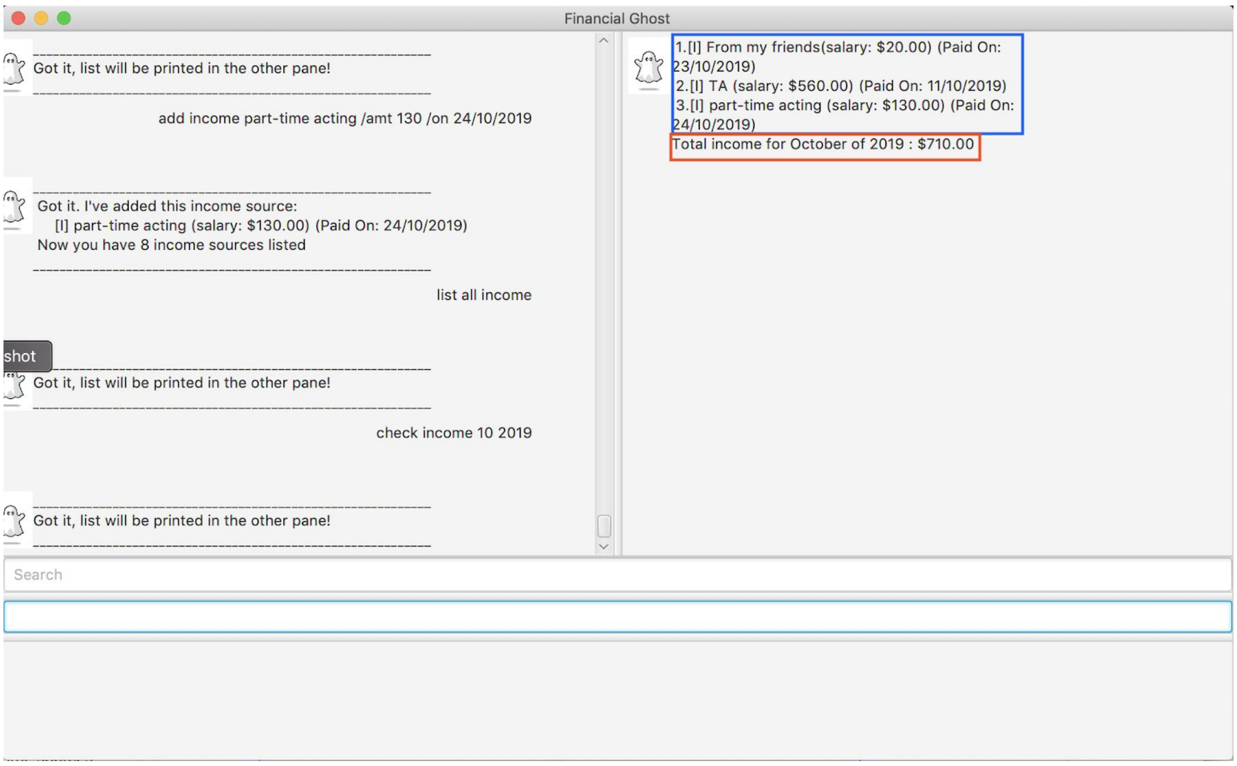
3.2.7 Check archive: `check income/expenditure`

Allows the user to check their income or expenditure records for previous months

Format:
- `check income [month] [year]`
- `check expenditure [month] [year]`

Example 1:
- User enters `check income 10 2019` into the command bar.

- The income list containing the income sources dated to the month of October 2019 will be displayed in the pane on the right as enclosed by the blue box
- The total monetary inflow from the income sources for that month is computed and displayed below the list as enclosed by the red box



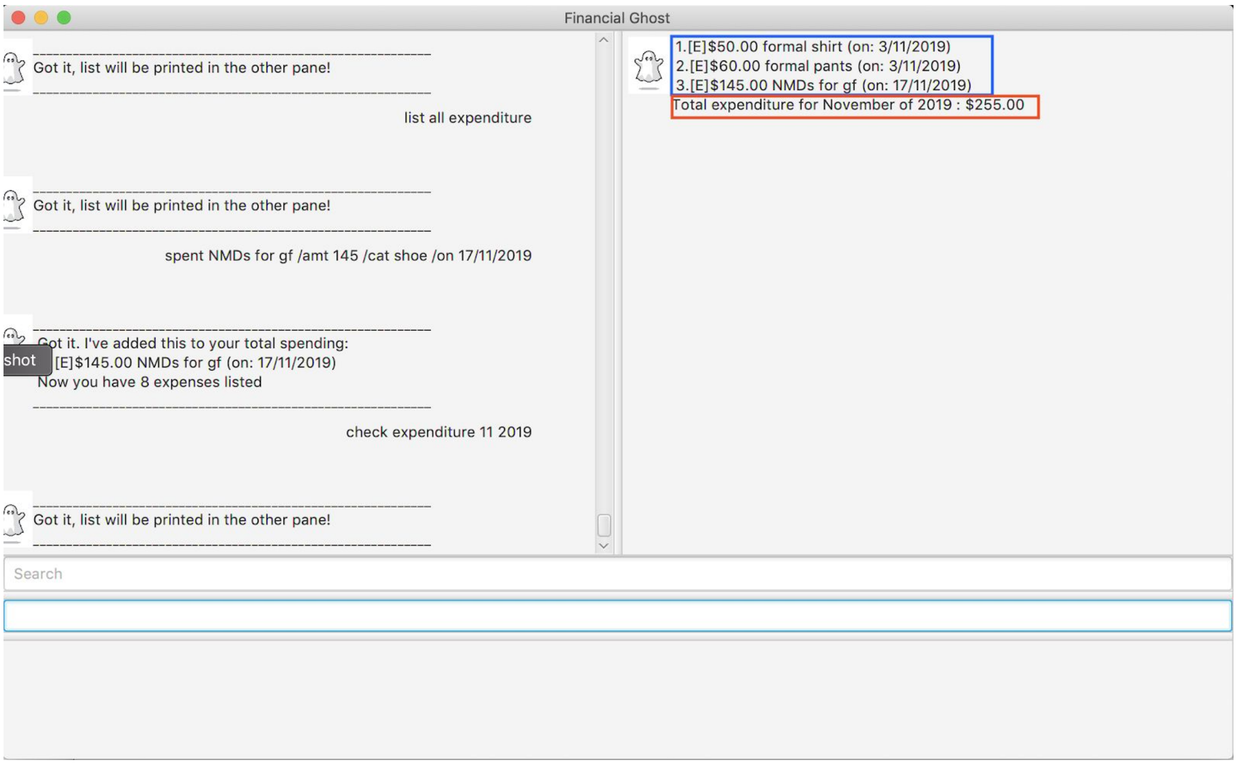
```
Financial Ghost

____________________________________________________________
Got it, list will be printed in the other pane!
____________________________________________________________
                         add income part-time acting /amt 130 /on 24/10/2019

____________________________________________________________
 Got it. I've added this income source:
    [I] part-time acting (salary: $130.00) (Paid On: 24/10/2019)
 Now you have 8 income sources listed
____________________________________________________________
                                                     list all income
shot
____________________________________________________________
Got it, list will be printed in the other pane!
____________________________________________________________
                                              check income 10 2019

____________________________________________________________
Got it, list will be printed in the other pane!
____________________________________________________________
```

```
1.[I] From my friends(salary: $20.00) (Paid On:
23/10/2019)
2.[I] TA (salary: $560.00) (Paid On: 11/10/2019)
3.[I] part-time acting (salary: $130.00) (Paid On:
24/10/2019)
Total income for October of 2019 : $710.00
```

Example 2:

- User enters `check expenditure 10 2019` into the command bar.

- The income list containing the expenditures dated to the month of September 2015 will be displayed in the pane on the right as enclosed by the blue box
- The total monetary outflow from the income sources for that month is computed and displayed below the list as enclosed by the red box


Financial Ghost

```
------------------------------------------------------------
Got it, list will be printed in the other pane!
------------------------------------------------------------
                                        list all expenditure
------------------------------------------------------------
Got it, list will be printed in the other pane!
------------------------------------------------------------
        spent NMDs for gf /amt 145 /cat shoe /on 17/11/2019
------------------------------------------------------------
Got it. I've added this to your total spending:
  [E]$145.00 NMDs for gf (on: 17/11/2019)
Now you have 8 expenses listed
------------------------------------------------------------
                                   check expenditure 11 2019
------------------------------------------------------------
Got it, list will be printed in the other pane!
------------------------------------------------------------
```

```
1.[E]$50.00 formal shirt (on: 3/11/2019)
2.[E]$60.00 formal pants (on: 3/11/2019)
3.[E]$145.00 NMDs for gf (on: 17/11/2019)
Total expenditure for November of 2019 : $255.00
```

3.2.8 Check Current Month: `list month income/expenditure`

Allows the user to check their income or expenditure records for the current month

Format:
- `list month income`
- `list month expenditure`

Example 1:
- User enters `list month income` into the command bar.
- The income list containing the income sources dated to the current month will be displayed in the pane on the right
- The total monetary inflow from the income sources for that month is computed and displayed below the list

Example 2:
- User enters `list month expenditure` into the command bar.
- The income list containing the expenditures dated to the current month will be displayed in the pane on the right
- The total monetary outflow from the expenditure for that month is computed and displayed below the list

3.3. Goal Setting

This section contains the commands to manage the user's financial goals.

Things that we want to buy, big or small, are always at the back of our heads when we do any sort of financial planning. This feature allows users to manage such goals in a systematic manner. It helps to do the financial planning for the user so that the user will be able to accomplish the goals that they have set for themselves.

3.3.1 Add goal: `goal`

Allow the user to track their goals by computing the recommended amount of savings per month to reach their goals.

Format: `goal [desc] /amt [cost] /by [d/m/yyyy] /priority [priority level]`

Examples:

- `goal buy HDB /amt 100000 /by 9/12/2030 /priority HIGH`
- `goal buy Lambo /amt 150000 /by 10/12/2040 /priority MEDIUM`
- `goal buy Boat /amt 200000 /by 11/12/2050 /priority LOW`

3.3.2 Delete goal: `delete goal`

Allows the user to delete the goals that they have set.

Format: `delete goal [index]`

Examples:

- `delete goal 1`

3.3.3 List and check goal progress: `list goals`

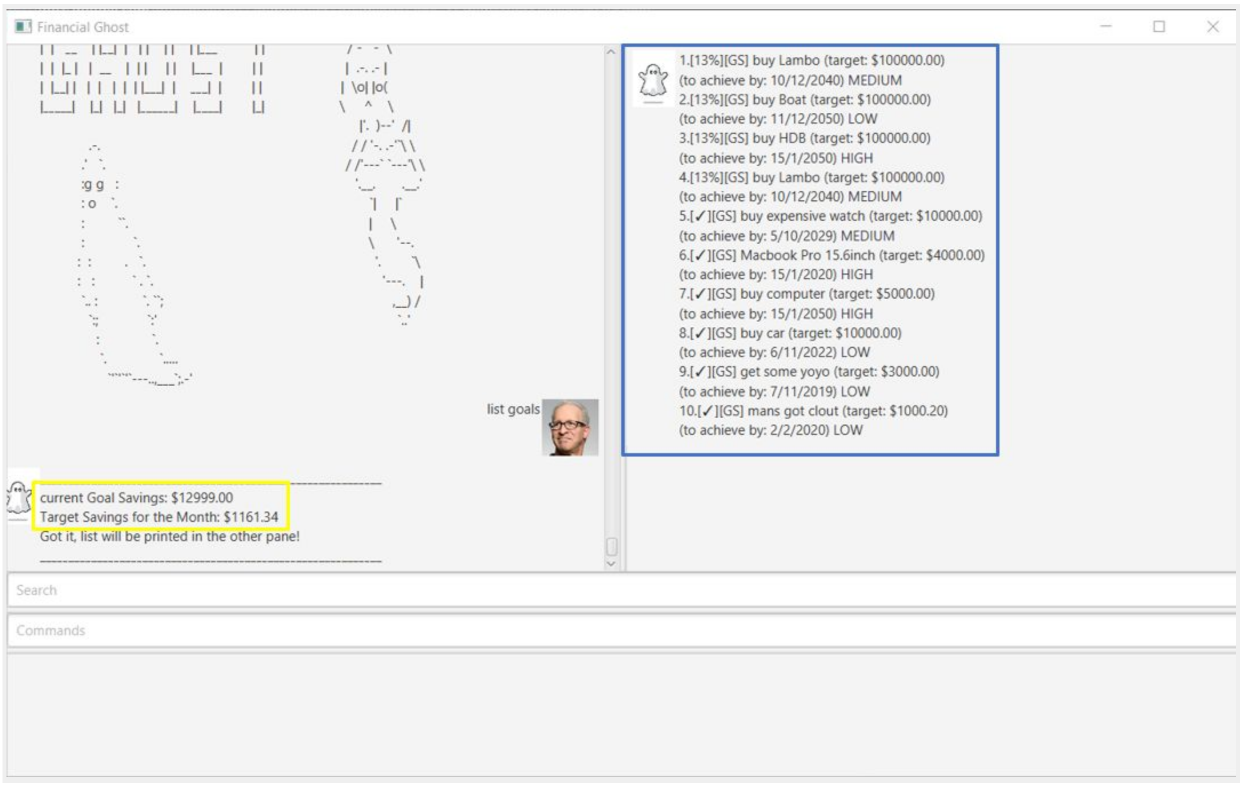
Lists and checks for the progress the user has made towards their set goals.

Format: `list goals`

Example:

1. User enters `list goals` into the command bar.
2. The goal list will be displayed in the pane on the right as enclosed by the blue box. It shows the progress the user has made towards their goals in square brackets e.g [13.5%]. If the user is able to afford to complete a goal, a tick will be displayed in square brackets.

3. The current goal savings and target savings for the month is displayed in the pane on the left as enclosed by the yellow box.



```
1.[13%][GS] buy Lambo (target: $100000.00)
(to achieve by: 10/12/2040) MEDIUM
2.[13%][GS] buy Boat (target: $100000.00)
(to achieve by: 11/12/2050) LOW
3.[13%][GS] buy HDB (target: $100000.00)
(to achieve by: 15/1/2050) HIGH
4.[13%][GS] buy Lambo (target: $100000.00)
(to achieve by: 10/12/2040) MEDIUM
5.[✓][GS] buy expensive watch (target: $10000.00)
(to achieve by: 5/10/2029) MEDIUM
6.[✓][GS] Macbook Pro 15.6inch (target: $4000.00)
(to achieve by: 15/1/2020) HIGH
7.[✓][GS] buy computer (target: $5000.00)
(to achieve by: 15/1/2050) HIGH
8.[✓][GS] buy car (target: $10000.00)
(to achieve by: 6/11/2022) LOW
9.[✓][GS] get some yoyo (target: $3000.00)
(to achieve by: 7/11/2019) LOW
10.[✓][GS] mans got clout (target: $1000.20)
(to achieve by: 2/2/2020) LOW
```

```
current Goal Savings: $12999.00
Target Savings for the Month: $1161.34
Got it, list will be printed in the other pane!
```

3.3.4 Complete set goal: `done goal`

Allows the user to mark their goal as complete and automatically converts goal into the current month’s expenditure.

Returns the remaining goals and shows the user the updated progress the user has made towards their goals.

Things to note:

- Goal will not be set to complete if the user does not have enough goal savings.

Format: `done goal [index]`
Example:

- `done goal 1`

3.3.5 Goal planner: `commit goal`

Allows the user to simulate the impact on his finances before he commits to spending money to complete their set goal.

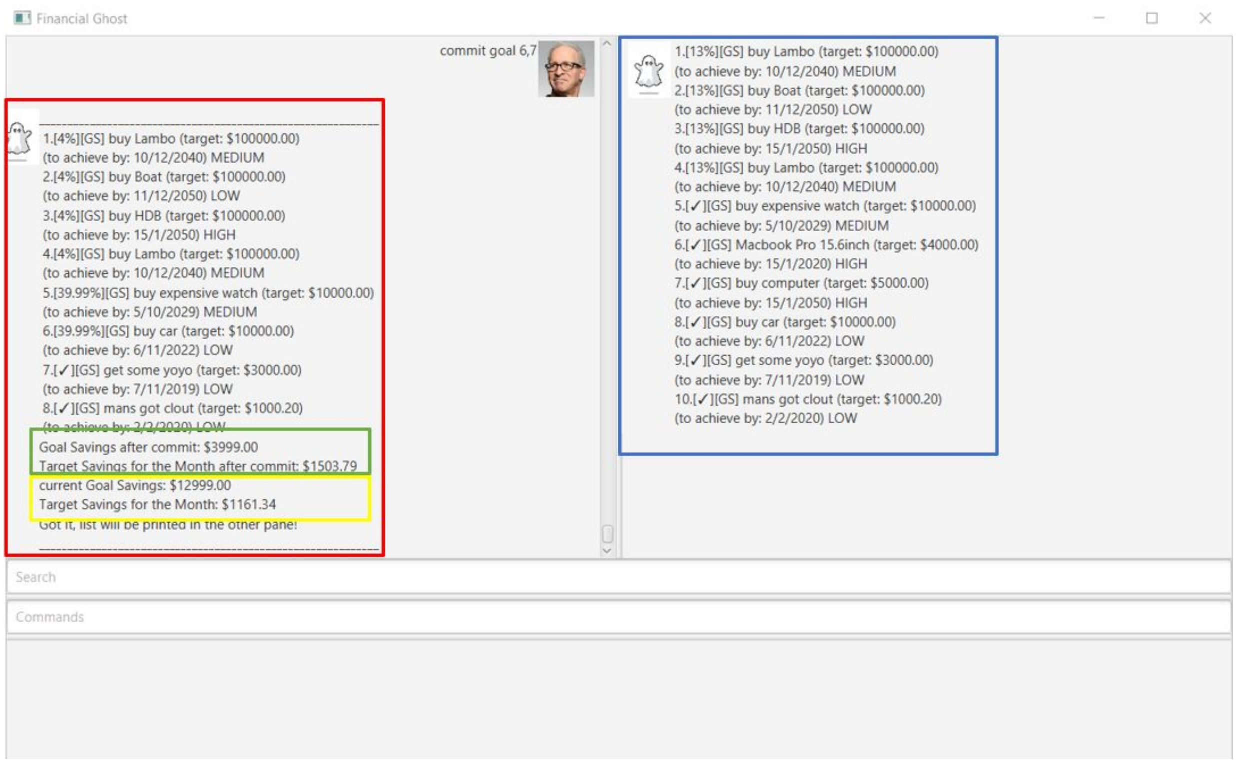
Returns the remaining goals after the simulated purchase. It also shows the user the updated progress of the goals remaining, goal savings and the target savings for the month after the simulated purchase.

Things to note:

- Goal will not be committed if the user does not have enough goal savings.
- Indexes input for the committed goals must be distinct.

Format: `commit goal [index 1, index 2, ...]`
Example:

1. User enters `commit goal 6,7` into the command bar.
2. The simulated goal list will be displayed in the pane on the left as enclosed by the red box, showing the updated progress towards the remaining goals in square brackets e.g [71.48%].
3. The simulated goal savings and target savings for the month are shown as enclosed by the green box.
4. The actual goal savings and target savings for the month is as shown in the yellow box.
5. The actual goal list is displayed in the pane on the right as enclosed by the blue box.



```
commit goal 6,7
```

```
--------------------------------------------------------------
1.[4%][GS] buy Lambo (target: $100000.00)
(to achieve by: 10/12/2040) MEDIUM
2.[4%][GS] buy Boat (target: $100000.00)
(to achieve by: 11/12/2050) LOW
3.[4%][GS] buy HDB (target: $100000.00)
(to achieve by: 15/1/2050) HIGH
4.[4%][GS] buy Lambo (target: $100000.00)
(to achieve by: 10/12/2040) MEDIUM
5.[39.99%][GS] buy expensive watch (target: $10000.00)
(to achieve by: 5/10/2029) MEDIUM
6.[39.99%][GS] buy car (target: $10000.00)
(to achieve by: 6/11/2022) LOW
7.[✓][GS] get some yoyo (target: $3000.00)
(to achieve by: 7/11/2019) LOW
8.[✓][GS] mans got clout (target: $1000.20)
(to achieve by: 2/2/2020) LOW
Goal Savings after commit: $3999.00
Target Savings for the Month after commit: $1503.79
current Goal Savings: $12999.00
Target Savings for the Month: $1161.34
Got it, list will be printed in the other pane!
--------------------------------------------------------------
```

```
1.[13%][GS] buy Lambo (target: $100000.00)
(to achieve by: 10/12/2040) MEDIUM
2.[13%][GS] buy Boat (target: $100000.00)
(to achieve by: 11/12/2050) LOW
3.[13%][GS] buy HDB (target: $100000.00)
(to achieve by: 15/1/2050) HIGH
4.[13%][GS] buy Lambo (target: $100000.00)
(to achieve by: 10/12/2040) MEDIUM
5.[✓][GS] buy expensive watch (target: $10000.00)
(to achieve by: 5/10/2029) MEDIUM
6.[✓][GS] Macbook Pro 15.6inch (target: $4000.00)
(to achieve by: 15/1/2020) HIGH
7.[✓][GS] buy computer (target: $5000.00)
(to achieve by: 15/1/2050) HIGH
8.[✓][GS] buy car (target: $10000.00)
(to achieve by: 6/11/2022) LOW
9.[✓][GS] get some yoyo (target: $3000.00)
(to achieve by: 7/11/2019) LOW
10.[✓][GS] mans got clout (target: $1000.20)
(to achieve by: 2/2/2020) LOW
```

3.4. Loan Management

The exchange of monetary loans is commonplace in the daily lives of Singaporeans. We regularly make both incoming and outgoing loans, such as borrowing from banks and lending money to friends or colleagues. This feature allows the user to track loans, both incoming and outgoing, and the status of the debt.

3.4.1 Add an incoming loan: `borrowed`

Allows the user to track incoming loans borrowed from organisations or other people

Adds an incoming loan to the loan list

Format: `borrowed [description] /amt [amount in dollars] /on d/m/yyyy`

Examples:

- `borrowed UOB /amt 50000 /on 4/3/2028`
- `borrowed NUS Student Loan /amt 4000 /on 6/7/2019`
- `borrowed my parents /amt 300 /on 17/12/2017`

3.4.2 Add an outgoing loan: `lent`

Allows the user to track outgoing loans lent to organisations or other people

Adds an outgoing loan to the loan list

Format: `lent [description] /amt [amount in dollars] /on d/m/yyyy`

Examples:

- `lent Sammi /amt 250 /on 5/4/2019`
- `lent Caleb-san /amt 100 /on 15/8/2019`
- `lent my brother /amt 700 /on 1/12/2017`

3.4.3 List loans according to type: `list all/incoming/outgoing loans`

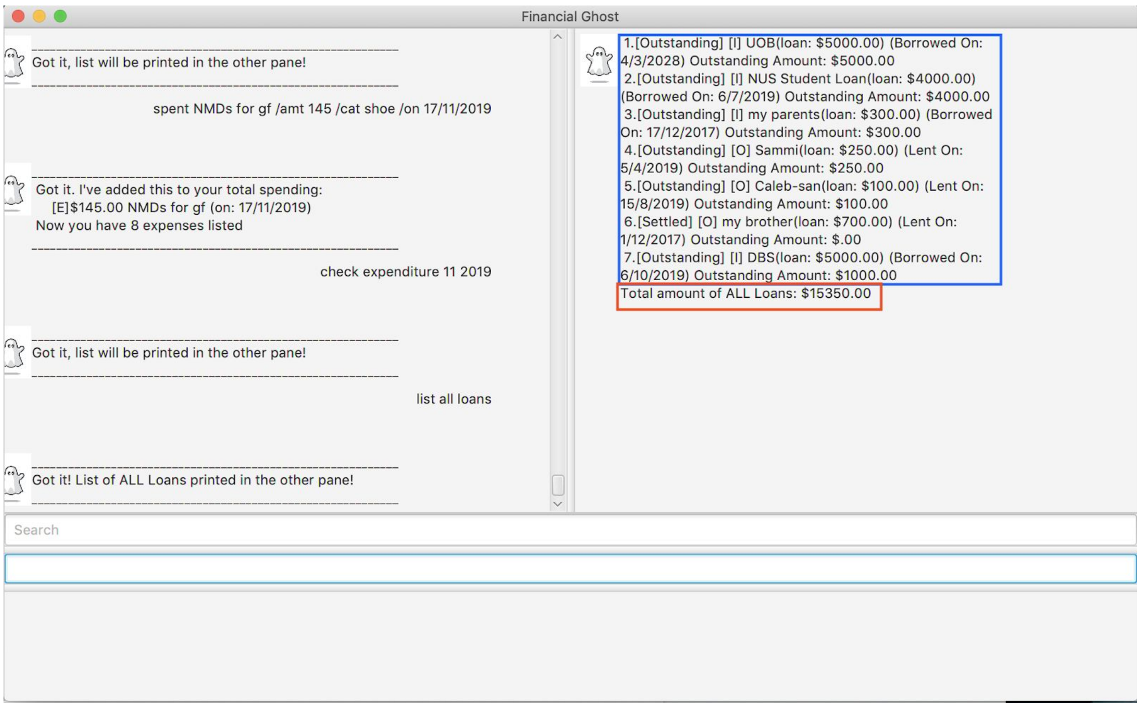
List loans present in the loans list according to type as specified by the user

Users may choose to list all incoming loans, all outgoing loans or both types of loans.

Format:
- `list all loans`
- `list outgoing loans`
- `list incoming loans`

Example:

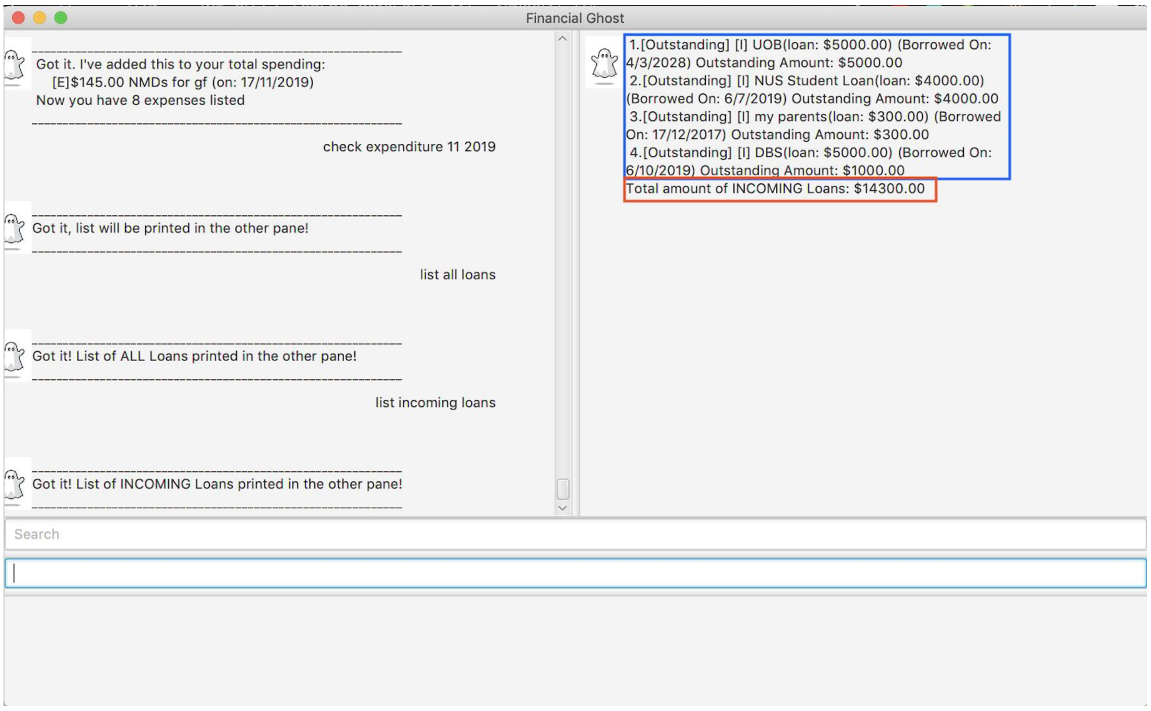
1. User enters `list all loans` into the command bar.
2. The loan list containing all loans will be displayed in the pane on the right as enclosed by the blue box
3. The total amount from all loans is computed and displayed below the list as shown in the red box.
4. User enters either `list outgoing loans` or `list incoming loans` to list incoming or outgoing loans
5. Likewise the loans are displayed in the right pane as enclosed by the blue box and the total amount from either the incoming or outgoing loans is shown in the red box.



```
Got it, list will be printed in the other pane!

spent NMDs for gf /amt 145 /cat shoe /on 17/11/2019

Got it. I've added this to your total spending:
    [E]$145.00 NMDs for gf (on: 17/11/2019)
Now you have 8 expenses listed

check expenditure 11 2019

Got it, list will be printed in the other pane!

list all loans

Got it! List of ALL Loans printed in the other pane!
```

```
1.[Outstanding] [I] UOB(loan: $5000.00) (Borrowed On: 4/3/2028) Outstanding Amount: $5000.00
2.[Outstanding] [I] NUS Student Loan(loan: $4000.00) (Borrowed On: 6/7/2019) Outstanding Amount: $4000.00
3.[Outstanding] [I] my parents(loan: $300.00) (Borrowed On: 17/12/2017) Outstanding Amount: $300.00
4.[Outstanding] [O] Sammi(loan: $250.00) (Lent On: 5/4/2019) Outstanding Amount: $250.00
5.[Outstanding] [O] Caleb-san(loan: $100.00) (Lent On: 15/8/2019) Outstanding Amount: $100.00
6.[Settled] [O] my brother(loan: $700.00) (Lent On: 1/12/2017) Outstanding Amount: $.00
7.[Outstanding] [I] DBS(loan: $5000.00) (Borrowed On: 6/10/2019) Outstanding Amount: $1000.00
Total amount of ALL Loans: $15350.00
```

All loans


Financial Ghost

Got it. I've added this to your total spending:
[E]$145.00 NMDs for gf (on: 17/11/2019)
Now you have 8 expenses listed

check expenditure 11 2019

Got it, list will be printed in the other pane!

list all loans

Got it! List of ALL Loans printed in the other pane!

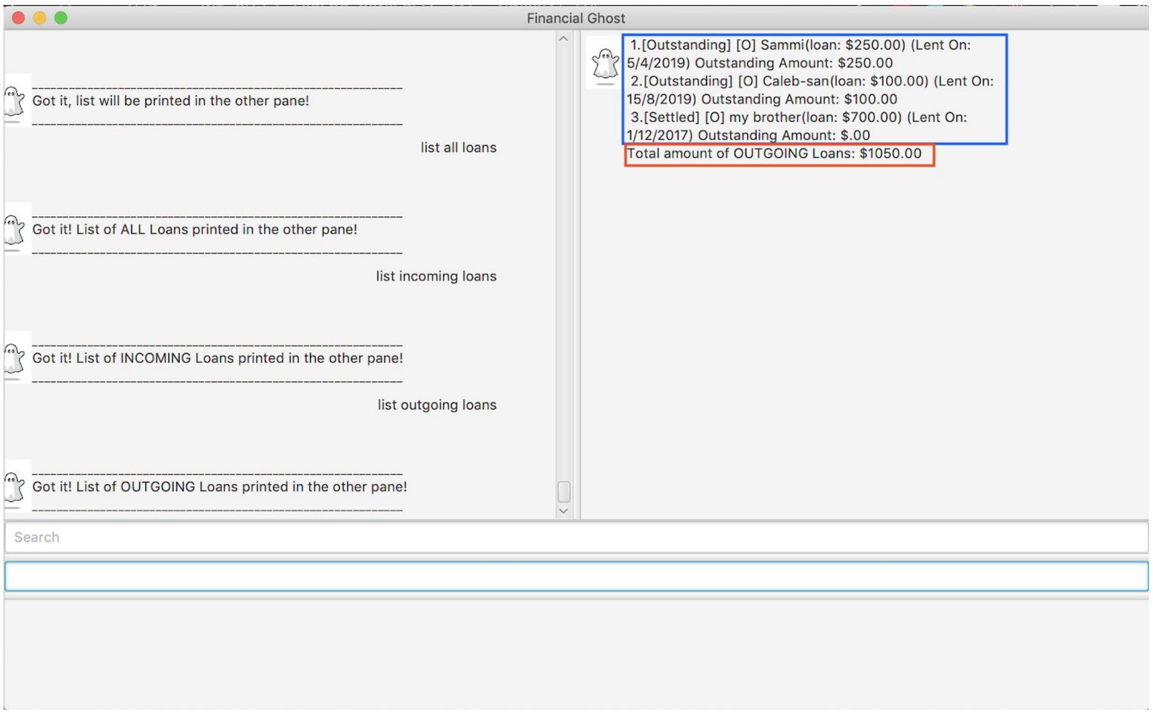
list incoming loans

Got it! List of INCOMING Loans printed in the other pane!

1.[Outstanding] [I] UOB(loan: $5000.00) (Borrowed On: 4/3/2028) Outstanding Amount: $5000.00
2.[Outstanding] [I] NUS Student Loan(loan: $4000.00) (Borrowed On: 6/7/2019) Outstanding Amount: $4000.00
3.[Outstanding] [I] my parents(loan: $300.00) (Borrowed On: 17/12/2017) Outstanding Amount: $300.00
4.[Outstanding] [I] DBS(loan: $5000.00) (Borrowed On: 6/10/2019) Outstanding Amount: $1000.00
Total amount of INCOMING Loans: $14300.00

Search

Incoming loans


Financial Ghost

Got it, list will be printed in the other pane!

list all loans

Got it! List of ALL Loans printed in the other pane!

list incoming loans

Got it! List of INCOMING Loans printed in the other pane!

list outgoing loans

Got it! List of OUTGOING Loans printed in the other pane!

1.[Outstanding] [O] Sammi(loan: $250.00) (Lent On: 5/4/2019) Outstanding Amount: $250.00
2.[Outstanding] [O] Caleb-san(loan: $100.00) (Lent On: 15/8/2019) Outstanding Amount: $100.00
3.[Settled] [O] my brother(loan: $700.00) (Lent On: 1/12/2017) Outstanding Amount: $.00
Total amount of OUTGOING Loans: $1050.00

Search

Outgoing loans

3.4.4 Settle an outgoing loan: `received`

Settles the debt of an outgoing loan according to the amount repaid

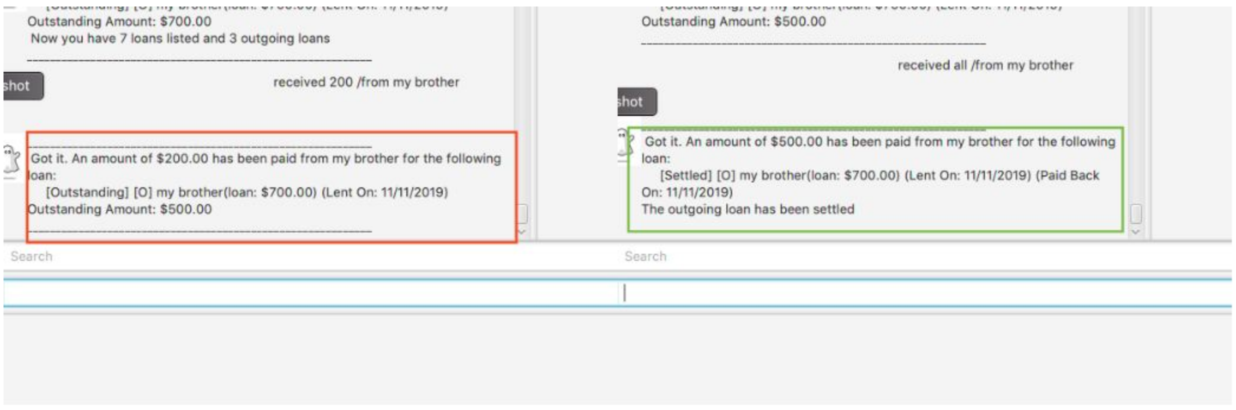
Amount repaid by the other party is automatically added to income

Format:

- `received [amount in dollars] /from [name of party]`
- `received [amount in dollars] /from [index of party]`
- `received all /from [name of party]`
- `received all /from [index of party]`

Examples:

1. User enters `received 200 /from my brother` into the command bar and press `Enter` on the keyboard
2. An amount of 200 is settled for the outgoing loan to my brother and deducted from the outstanding amount of the loan
3. The amount of 200 is added automatically to the total income list
4. The success message will be shown on the left pane as enclosed by the red box.
5. User enters `received all /from my brother` into the command bar and press `Enter` on the keyboard to settle the remaining debt
6. An amount of 500 is settled for the outgoing loan and the loan is set to Settled
7. The amount of 500 is added automatically to the total income list
8. The success message will be shown on the left pane as enclosed by the green box.



```
Outstanding Amount: $700.00
 Now you have 7 loans listed and 3 outgoing loans
----------------------------------------------------------------
                                   received 200 /from my brother

----------------------------------------------------------------
Got it. An amount of $200.00 has been paid from my brother for the following
loan:
    [Outstanding] [O] my brother(loan: $700.00) (Lent On: 11/11/2019)
Outstanding Amount: $500.00
----------------------------------------------------------------
```

```
Outstanding Amount: $500.00
----------------------------------------------------------------
                                   received all /from my brother

Got it. An amount of $500.00 has been paid from my brother for the following
loan:
    [Settled] [O] my brother(loan: $700.00) (Lent On: 11/11/2019) (Paid Back
On: 11/11/2019)
The outgoing loan has been settled
```

3.4.5 Settle an incoming loan: `paid`

Settles the debt of an incoming loan according to the amount repaid

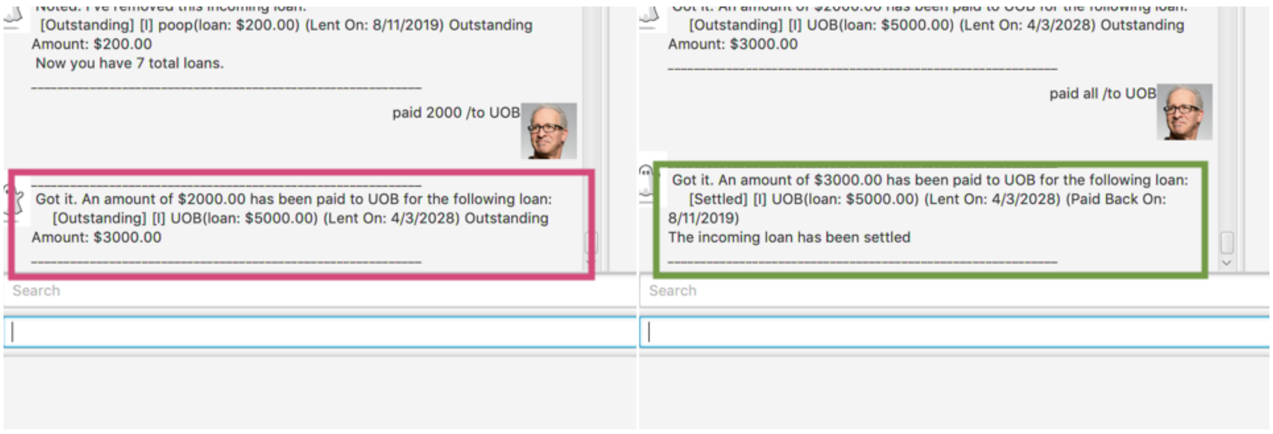
Amount repaid to the other party is automatically added to expenditure

Format:

- `paid [amount in dollars] /to [name of party]`
- `paid [amount in dollars] /to [index of party]`
- `paid all /to [name of party]`
- `paid all /to [index of party]`

Examples:

1. User enters `paid 2000 /to UOB` into the command bar and presses `Enter` on the keyboard
2. An amount of 2000 is settled for the incoming loan from UOB and deducted from the outstanding amount of the debt
3. The amount of 2000 is automatically added to the total expenditure list
4. The success message will be shown on the left pane as enclosed by the red box
5. User enters `paid all /to UOB` into the command bar and presses `Enter` on the keyboard to settle the remaining debt
6. An amount of 48000 is settled for the incoming loan and the status of the loan is set to Settled
7. The amount of 48000 is automatically added to the total expenditure list
8. The success message will be shown on the left pane as enclosed by the green box



```
 [Outstanding] [I] poop(loan: $200.00) (Lent On: 8/11/2019) Outstanding
Amount: $200.00
 Now you have 7 total loans.
____________________________________________________________

                                            paid 2000 /to UOB
____________________________________________________________
 Got it. An amount of $2000.00 has been paid to UOB for the following loan:
    [Outstanding] [I] UOB(loan: $5000.00) (Lent On: 4/3/2028) Outstanding
Amount: $3000.00
____________________________________________________________
Search
```

```
    [Outstanding] [I] UOB(loan: $5000.00) (Lent On: 4/3/2028) Outstanding
Amount: $3000.00
____________________________________________________________

                                            paid all /to UOB
 Got it. An amount of $3000.00 has been paid to UOB for the following loan:
    [Settled] [I] UOB(loan: $5000.00) (Lent On: 4/3/2028) (Paid Back On:
8/11/2019)
The incoming loan has been settled
____________________________________________________________
Search
```

3.4.6 Delete loan: `delete loan`

Allows the user to delete a loan that they have added according to the index of the loan in the loan list

Deletes both incoming and outgoing loans

Format: `delete loan [index]`

Examples:

- `delete loan 1`

Note: Index in the format refers to the index of the loan within the loan list, not the incoming loan list or outgoing loan list

3.5. Instalment Management

This section contains the commands to manage the user's outstanding instalments.

3.5.1 Add instalment: `add instalment`

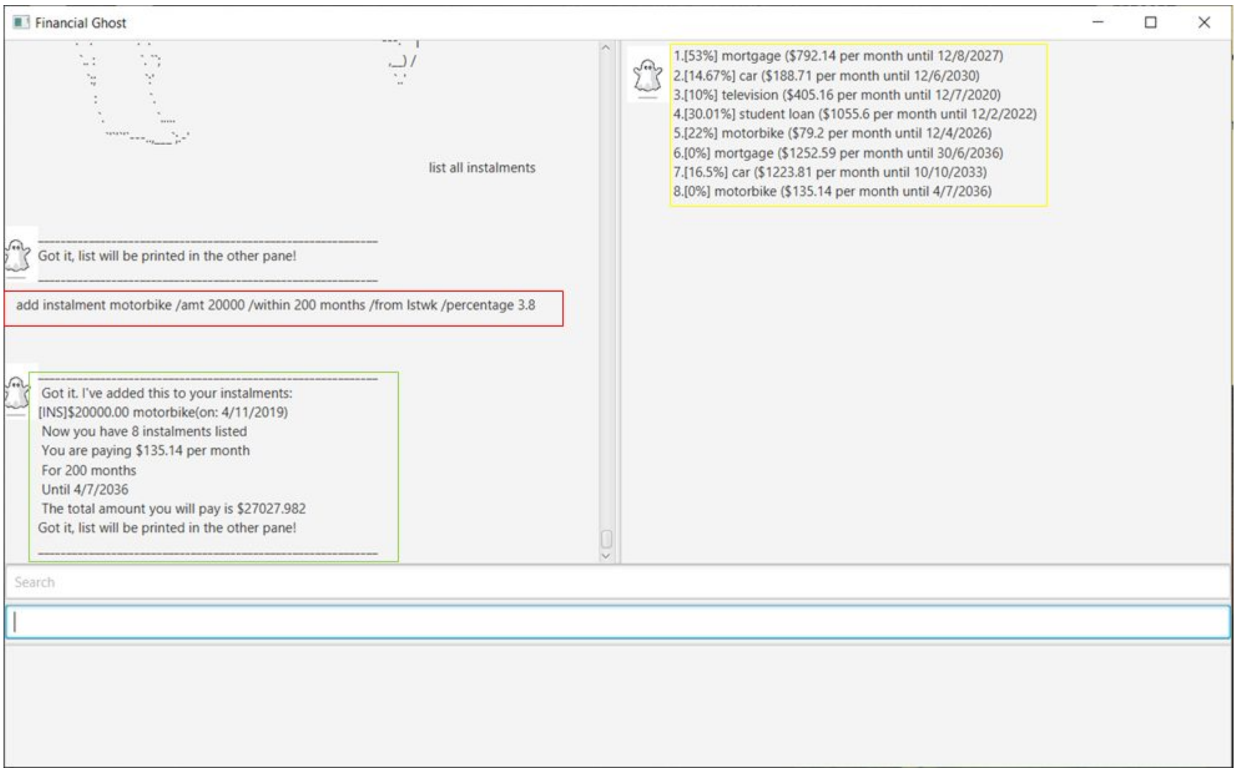
Allows the user to keep track of the uncompleted payments that they have to settle via recurring monthly instalments. Also shows a percentage bar at the same time.

Format: `add instalment [desc] /amt [cost] /within [number of months] months /from [d/m/yyyy] /percentage [annual interest rate]`

- `The right output will list all the instalments with the added instalment.`
- `The left output will show a success message.`

Examples:

1. User enters `add instalment motorbike /amt 20000 /within 200 months /from lstwk /percentage 3.8` into the command bar and press `Enter` on the keyboard.
2. The instalment list will be displayed in the pane on the right as enclosed by the yellow box, showing the progress towards the respective instalments in square brackets e.g [71.48%].
3. The success message will be shown on the left pane as enclosed by the green box.



```
list all instalments

Got it, list will be printed in the other pane!

add instalment motorbike /amt 20000 /within 200 months /from lstwk /percentage 3.8

Got it. I've added this to your instalments:
[INS]$20000.00 motorbike(on: 4/11/2019)
Now you have 8 instalments listed
You are paying $135.14 per month
For 200 months
Until 4/7/2036
The total amount you will pay is $27027.982
Got it, list will be printed in the other pane!
```

```
1.[53%] mortgage ($792.14 per month until 12/8/2027)
2.[14.67%] car ($188.71 per month until 12/6/2030)
3.[10%] television ($405.16 per month until 12/7/2020)
4.[30.01%] student loan ($1055.6 per month until 12/2/2022)
5.[22%] motorbike ($79.2 per month until 12/4/2026)
6.[0%] mortgage ($1252.59 per month until 30/6/2036)
7.[16.5%] car ($1223.81 per month until 10/10/2033)
8.[0%] motorbike ($135.14 per month until 4/7/2036)
```

3.5.2 Delete Instalment: `delete instalment`

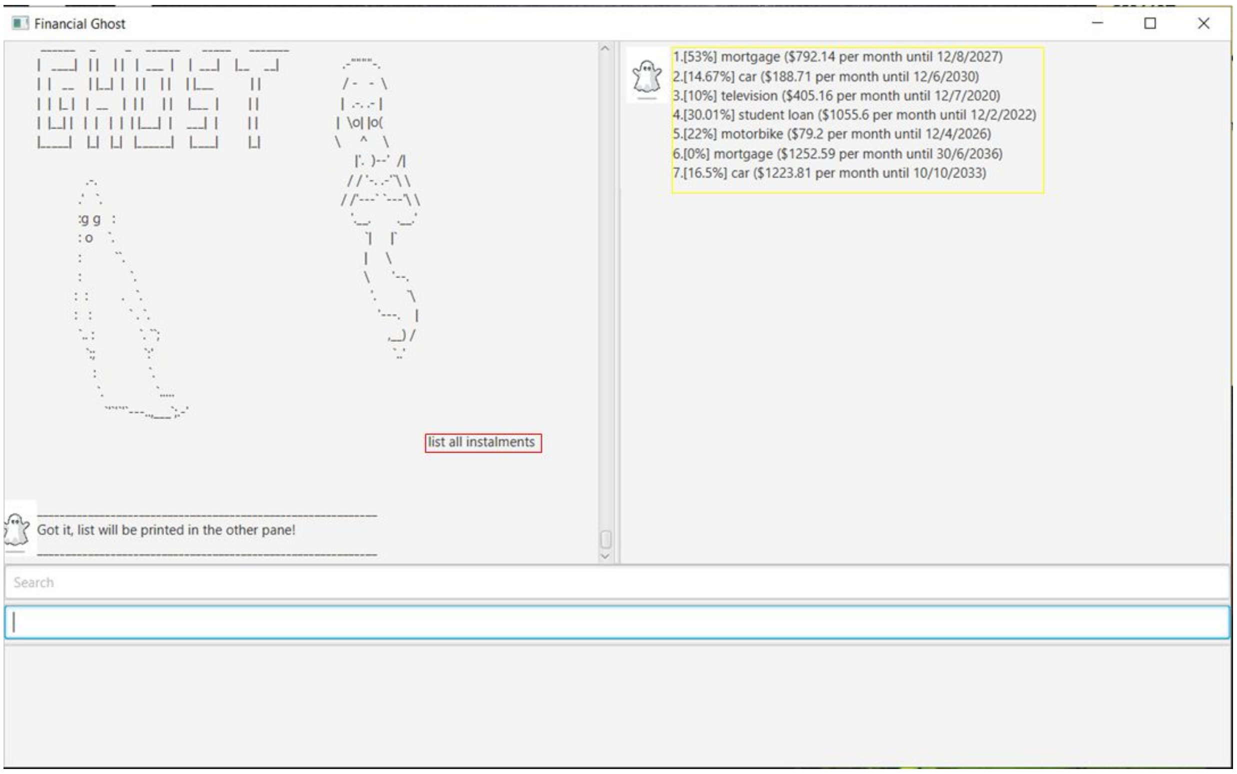
Allows the user to override the system and remove any instalment to account for circumstances when user defaults.

Format: `delete instalment [index]`

- `The right output will list all the instalments without the deleted instalment.`
- `The left output will show a success message.`

Example:

1. User enters `delete instalment 3` into the command bar and press `Enter` on the keyboard.
2. The instalment list will be displayed in the pane on the right as enclosed by the yellow box, showing the progress towards the respective instalments in square brackets e.g [71.48%].
3. The success message will be shown on the left pane as enclosed by the green box.



```
Got it, list will be printed in the other pane!
```

```
1.[53%] mortgage ($792.14 per month until 12/8/2027)
2.[14.67%] car ($188.71 per month until 12/6/2030)
3.[10%] television ($405.16 per month until 12/7/2020)
4.[30.01%] student loan ($1055.6 per month until 12/2/2022)
5.[22%] motorbike ($79.2 per month until 12/4/2026)
6.[0%] mortgage ($1252.59 per month until 30/6/2036)
7.[16.5%] car ($1223.81 per month until 10/10/2033)
```

3.5.3 List Instalments: `list all instalments`

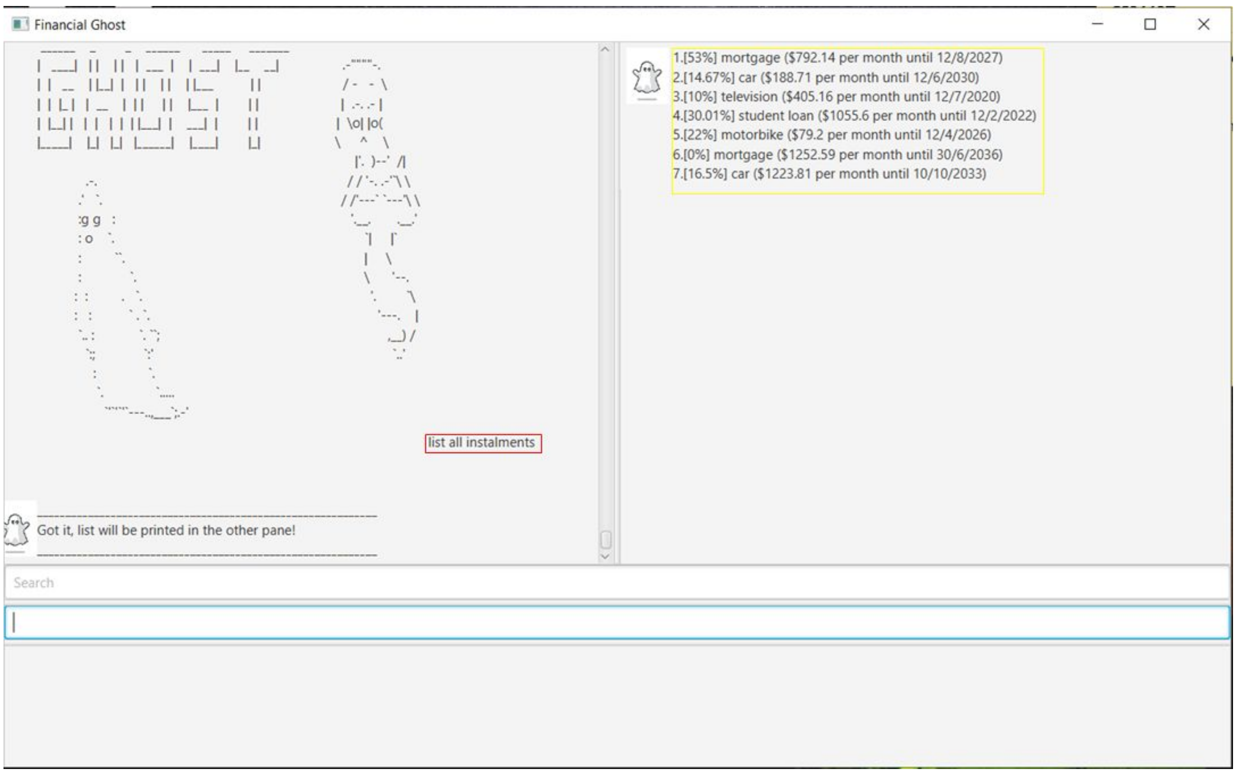
Allows the user to check all of the uncompleted instalments that they have to settle via recurring monthly instalments. Also shows a percentage bar at the same time.

Format: `list all instalments`

- `The output will list all the instalments.`

Example:

1. User enters `list all instalments` into the command bar.
2. The instalment list will be displayed in the pane on the right as enclosed by the yellow box. It shows the percentage the user has made towards their instalments in square brackets e.g [13.5%].



```
Got it, list will be printed in the other pane!
```

```
1.[53%] mortgage ($792.14 per month until 12/8/2027)
2.[14.67%] car ($188.71 per month until 12/6/2030)
3.[10%] television ($405.16 per month until 12/7/2020)
4.[30.01%] student loan ($1055.6 per month until 12/2/2022)
5.[22%] motorbike ($79.2 per month until 12/4/2026)
6.[0%] mortgage ($1252.59 per month until 30/6/2036)
7.[16.5%] car ($1223.81 per month until 10/10/2033)
```

3.5.4 Recurring Instalments

Automatically count monthly instalments as a part of Expenditure until last instalment and from there on it will terminate. Notify the user monthly when it deducts and when it ends. Auto remove it when it is done. No command required from the user.

3.6. Bank Account Management

This section contains the commands to manage and track the user’s bank accounts.

3.6.1 Create A Bank Account Tracker `bank-account`

Allows the user to create a bank account tracker to track the balance and interest rate of the bank account.

Format: `bank-account [description] /amt [initial amount of money] /at [initial date] /rate [interest rate]`

- Create a bank account tracker to track and manage the bank accounts
- Duplicate bank account description is now allowed, the exception will be thrown
- Bank account description is case-sensitive
- Extra blank spaces may bring errors
- Initial date format should be dd/MM/yyyy

Examples:

- `bank-account OCBC /amt 0 /at 27/7/2017 /rate 0.005`
- `bank-account DBS /amt 100 /at 14/8/2018 /rate 0`

3.6.2 Delete A Bank Account Tracker: `Delete bank-tracker`

Allows the user to delete the bank tracker they do not want based on the index of the bank account tracker in the list.

Format: `delete bank-account [index of the bank account tracker]`

- The index should be a positive number and small then the total number of trackers currently existing.
- The user may need to use list command to check the index of the tracker first.

Examples:

- `delete bank-account 1`
- `delete bank-account 3`

3.6.3. List Bank Account Trackers: `list bank trackers`

Allows the user to check all the bank account trackers recorded in Financial Ghost.

Format: `list bank trackers`

- `The index of the tracker in the list can be used to run the delete bank-tracker command`

3.6.4. Internal Transfer: `deposit/withdraw`

Allows the user to keep track of deposit and withdraw money. When the user deposit money, the money will also be counted as a new income and same for withdrawing the money.

Format: `deposit/withdraw [amount] [account_description] /at [date]`

- `The amount should be a positive number`
- `The date should be after the latest update date of the tracker`
- `Bank account description is case-sensitive`
- `The amount should not be a negative number`
- `The date should follow the format dd/MM/yyyy`

Examples:

- `deposit 200 OCBC /at 28/9/2019`
- `withdraw 50 DBS /at 25/9/2019`

3.6.5. Check balance: `check-balance`

Allows the user to check the balance in their accounts at a specific date after the initial date based on the current statistics.

Format: `check-balance [account_description] /at [future_date]`

- `The [future_date] must be after the latest update date of the account tracker`
- `Bank account description is case-sensitive`
- `Future date should follow dd/MM/yyyy`

Examples:

- `check-balance OCBC /at 12/3/2020`

3.7. Undo

The user is able to undo up to 5 latest commands they have issued to the program.

Format: `undo`

Entering undo reverts the program to its previous state before the command was implemented. Only the following types of commands can be undone by the undo command:

- Commands that add to the account database
- Commands that delete an entry from the account database
- Commands that alter an entry from the account database

Commands like list or graph that do not involve making changes to the database will print an error message: “Command can’t be undone!”.

If program prints message: “No commands to undo!” it could mean that there were no commands prior to the undo command were issued to the program, or undo has been called more than 5 times in succession.

Details on the types of commands that can and cannot be undone is found in the Command Summary (Section 6).

3.8. Search

The user is able to search for items grouped according to their respective types. The search contents are to be entered in the search bar. The search results will be updated upon every key press. This is to allow the user to quickly search for an item in the system for their reference.

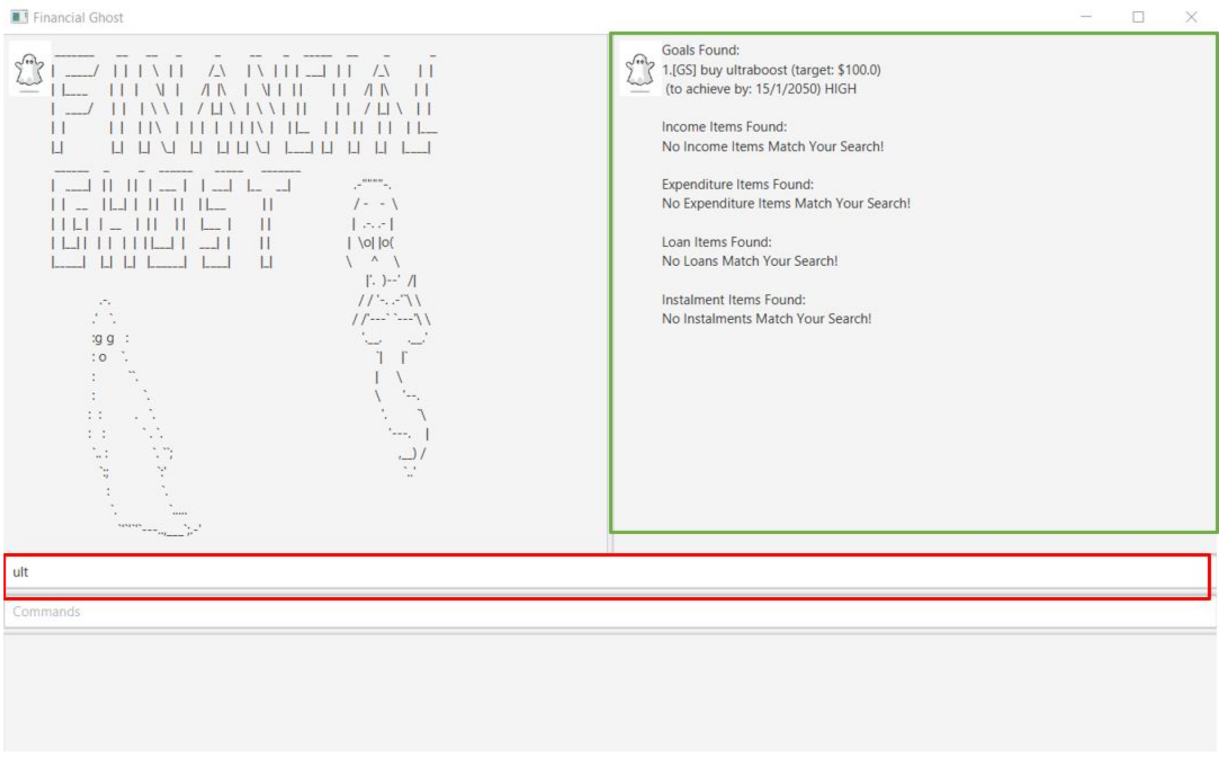
Things to note:

- The search results are case sensitive.
- Only Alphanumeric Search Inputs are Supported

Format: [search contents]

Example:

1. [search contents] are to be entered in the search bar as enclosed by the red box.
2. The search results are shown in the pane on the right as enclosed by the green box.
3. The search results will appear as the user types in the search bar.


Financial Ghost

Goals Found:
1.[GS] buy ultraboost (target: $100.0)
(to achieve by: 15/1/2050) HIGH

Income Items Found:
No Income Items Match Your Search!

Expenditure Items Found:
No Expenditure Items Match Your Search!

Loan Items Found:
No Loans Match Your Search!

Instalment Items Found:
No Instalments Match Your Search!

ult

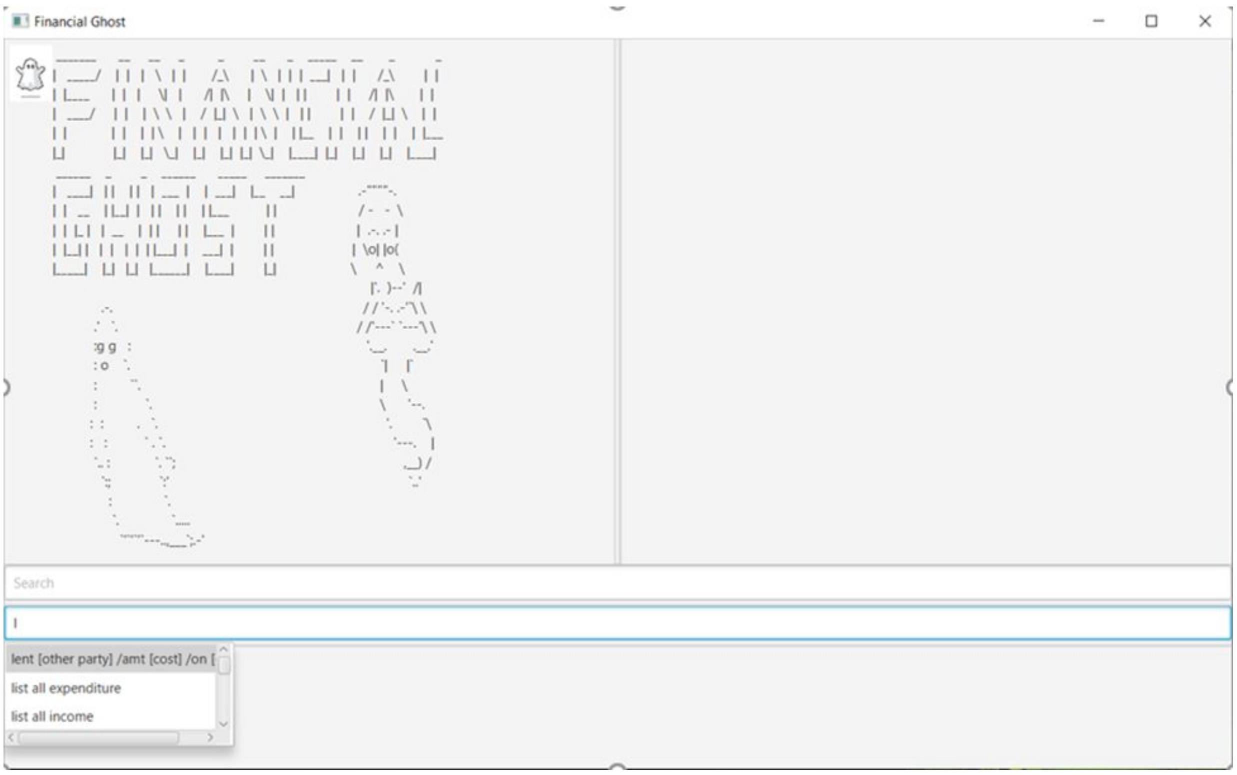
3.9. Help

This section comprises of user friendly features to help the user to adapt to the usage of Financial ghost.

3.9.1 Auto-Complete

This function autocompletes user commands using a drop down list.
With the help of the drop-down list of suggestions, you do not need to memorise any command. When you type your commands inside the command box, you will notice that a drop-down list appears.

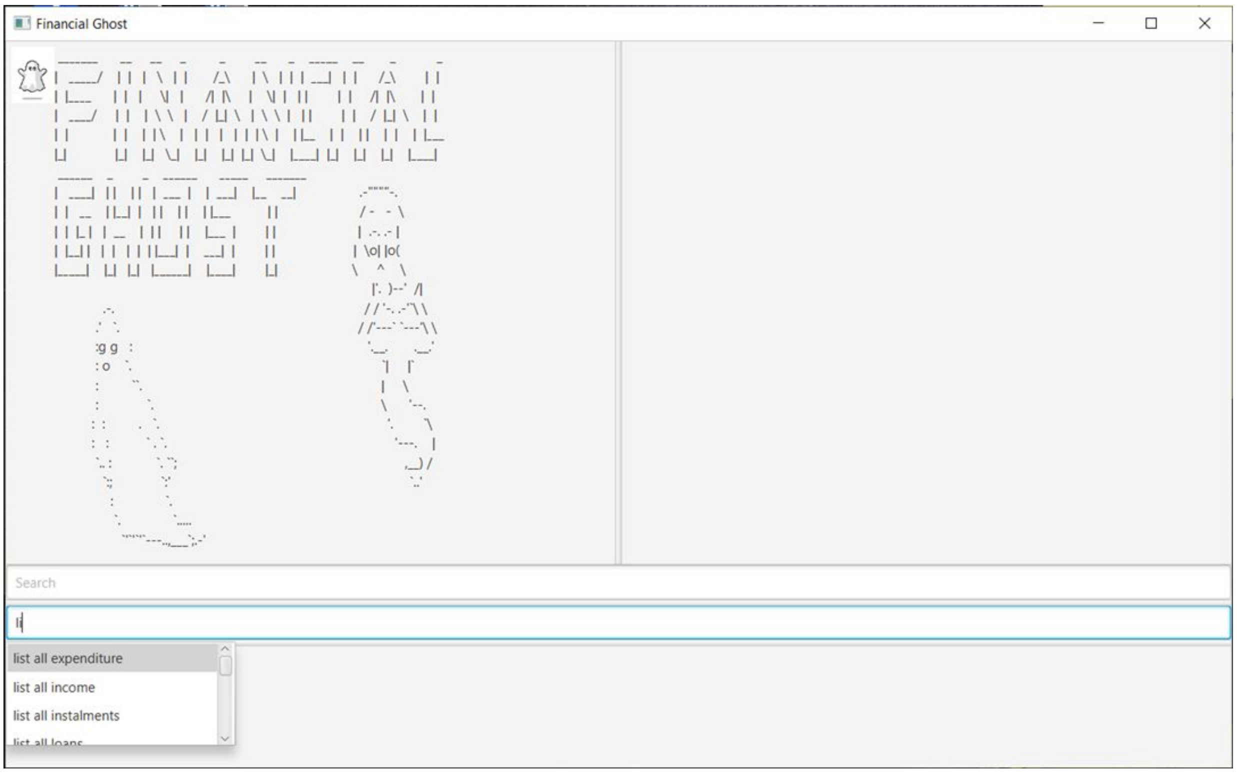
Example: If you type the letters “`l`", a drop-down list consisting of all options that starts with “`l`” appears, as shown in the screenshot below.



```
l
```

```
lent [other party] /amt [cost] /on [
list all expenditure
list all income
```

The list will get updated as you type, to only display commands that match your input. Following the previous example, if you type the letter "`i`" after having typed in "`l`", the list should only consist of the command that starts with “`li`” now.



```
li
```

```
list all expenditure
list all income
list all instalments
list all loans
```

When the drop-down list is shown, you can:

- Use `↑` and `↓` arrow keys to navigate through the list and press `Enter` to select the highlighted option, or
- Use your mouse to click on the option to select it

After you have selected the command you wish to enter, you should see it displayed in the command box.

Now you can:

- Edit those inside the "`[]`" to fill up the command format and remove "`[]`" at the same time. Press `Enter` to execute the command.
- Execute the command if there is no such "`[]`" in the command.

3.9.2 History

This function memorises previous commands entered by the user in the command text bar. This function acts on the `↑` and `↓` arrow keys. Hence, if a drop down list is shown as mentioned in section 3.9.2, this function cannot be used, else user can use the `↑` and `↓` arrow keys to navigate through the previous command that they have entered.

3.10. Timing Shortcuts `now` `ytd` `tmr` `lstwk` `nxtwk` `lstmth` `nxtmth` `lstyr` `nxtyr`

We recognised that users would often consider dates relative to the current date. In addition, inputting the date within each command in the format of `d/m/yyyy` could be a hassle.

Hence, for the sake of convenience, the user may choose to utilise timing shortcuts in place of dates in their commands.

Format:

- `now` (Shortcut for the current date)
- `ytd` (Shortcut for yesterday's date)
- `tmr` (Shortcut for tomorrow's date)
- `lstwk` (Shortcut for last week's date)
- `nxtwk` (Shortcut for next week's date)
- `lstmth` (Shortcut for last month's date)

- `nxtmth` (Shortcut for next month’s date)
- `lstyr` (Shortcut for last year’s date)
- `nxtyr` (Shortcut for next year’s date)

Examples:

- `add income Tuition /amt 1500 /on now`
- `lent Tom Cruise /amt 2000 /on ytd`
- `borrowed parents /amt 300 /on tmr`
- `goal Motorbike /amt 50000 /by nxtyr /priority HIGH`
- `add instalment apartment /amt 2000 /with 12 /from nxtmth /percentage 3`
- `bank-account UOB savings /amt 5000 /at nxtwk /rate 1`
- `spent ultra boost /amt 175 /cat shoes /on lstwk`

3.11. Statistical Reports

3.11.1. Financial Report for The Current Month: `graph monthly report`

Shows the total amount of income and expenditure for the current month.

Format: `graph monthly report (pie_chart/histogram/line_graph)`

- `(pie_chart/histogram/line_graph) can be omitted then Financial Ghost will show the default graph for the monthly report which is histogram.`
- `Besides pie_chart, histogram or line_graph, other patterns does not work and the default pattern (histogram) will be shown.`

Examples:

- `graph monthly report`
- `graph monthly report pie_chart`

3.11.2. Income/Expenditure Categories: `graph income/expenditure trend`

Shows a graph of the distribution of income/expenditure based on the type.

Format: `graph income/expenditure trend (pie_chart/histogram/line_graph)`

- `(pie_chart/histogram/line_graph) can be omitted then Financial Ghost will show the default graph for the category trend which is line graph.`
- `Besides pie_chart, histogram or line_graph, other patterns does not work and the default pattern (line graph) will be shown.`

Examples:

- `graph income trend`

- `graph expenditure trend histogram`

3.11.3. Graph for Financial Status: `graph financial status`

Shows a histogram of near 3 months' income and expenditure until the given date.

Format: `graph finance status /until [date]`

- `The date should follow the format dd/MM/yyyy.`
- `If there is no record for the 3 months, an empty histogram will be shown.`
- `Only 1 type of graph available for this command.`

Examples:

- `graph finance status /until 12/12/2019`
- `graph finance status /until 5/7/2019`

3.12. Exiting the program: `bye`

Exits the program.

Format: `bye`

3.13. Saving the file

Financial Ghosts writes data to the hard disk automatically after any command that changes the data.

There is no need to save manually.

3.14. Change Icon: `change icon`

Allow the user to upload their desired image as the user icon and store that image.

Format: `change icon`

- `Please place jar file under the path with only English words. The path with Non-English words (eg. Chinese characters) will make the program fail to load the image.`
- `If you change the location of the jar file after running it, this feature will not work.`
- `When the user restarts the program, it will load the latest saved version of the image as the user icon`

4. Features [coming in 2.0]

Below are future implementations that we Plan to release in version 2.0.

4.1. Login: `login`

The user needs password to access this software.

- If it is the first time to use Financial Ghost, the software will require the user to set up his/her account.
- After the first login, the password will be required to access Financial Ghost.

Examples:

- `[output] What is the password?`

4.2. Reminders for bills/instalments

UI gives a reminder to pay outstanding bills or instalments in a week's time before its deadline at the start of each session of the program.

- When registering a payment for instalments or bills, Financial Ghost will prompt the user to key in the next date which the next month's instalments or bills would be due.
- Reminders will not be shown if no outstanding payments to be made.
- Gives users the option to configure these reminders.

Example:

```
[User] starts up the program
[UI] <Prints greeting message>
       You have 2 outstanding payment(s) to be made:
          1.) Down payment for house (Due: 12/12/2019)
          2.) Utility bills (Due:14/12/2019)
Today's date: 7/12/2019
```

4.3. Fix-time deposit: `time-deposit`

Allows the user to keep track of time deposit with different interest rates. Financial Ghost will record and also calculate the amount of money in the future.

Format: `time-deposit [amount] [account_description] /from [start_date] /to [end_date]`

Examples:

- `time-deposit 200 OCBC /from 28/9/2019 /to 28/3/2020`
- `time-deposit 50 DBS /from 25/9/2019 /to 25/3/2020`

4.4. Monthly in-out trend: `trend`

Displays histogram of the financial statement for the selected year while displaying the average amount of money in and money out for the past 12 months.

Format: `trend [year]`

- The `[year]` must be a year after 2010.

Examples:

- `trend 2019`

4.5. Suggested Corrected Exception

When the input string cannot be recognized, an exception will be thrown and the error messages will be prompted in order to guide the user.

4.6. Help Page

General Help page. When the user types in the wrong command. Exception will be thrown to them and suggest them to look up the help page.

Format: `helppage`

- Throws out a help page.

4.7. Making autocomplete faster

Current implementation of drop down list of suggestions uses the third party library Controlsfx from Maven Central. Implementation seems to be a little slow. Will write an implementation directly instead of using third party library to make the drop down list appear faster.

5. FAQ

Q: How do I fix the data file in the case that the file becomes corrupted?

A: Navigate to the root directory of Financial Ghost folder located in your computer. Go the the data file located in the filepath: FinancialGhost/data/moneyAccount.txt and delete the line containing the error displayed in the error message.

6. Command Summary

Command Type	Format	Can be Undone (Y/N)
Initialise	`init [existing savings] [Avg Monthly Expenditure]`	N
Add Income	`add income [desc] /amt [amount in dollars] /on d/m/yyyy`	Y
Delete Income	`delete income [index]`	Y
List Full Income History	`list all income`	N
List Current Month Income History	`list month income`	N
List Past Month Income History	`check income [month] [year]`	N
Add Expenditure	`spent [desc] /amt [amount in dollars] /cat [category] /on d/m/yyyy`	Y
Delete Expenditure	`delete expenditure [index]`	Y
List Full Expenditure History	`list all expenditure`	N
List Current Month Expenditure History	`list month expenditure`	N
List Past Month Expenditure History	`check expenditure [month] [year]`	N

Add Goal	`goal [desc] /amt [cost] /by [d/m/yyyy] /priority [priority level]`	Y
Delete Goal	`delete goal [index]`	Y
List Goals	`list goals`	N
Complete Goal	`done goal [index]`	Y
Commit Goal	`commit goal [index 1, index 2,...]`	N
Add Instalment	`add instalment [desc] /amt [cost] /within [number of months] months /from [d/m/yyyy] /percentage [annual interest rate]`	Y
Delete Instalment	`delete instalment [index]`	Y
List Instalments	`list all instalments`	N
Add Incoming Loan	`lent [other party] /amt [cost] /on d/m/yyyy`	Y
Add Outgoing Loan	`borrowed [other party] /amt [cost] /on d/m/yyyy`	Y
Delete Loan	`delete loan [index]`	Y
List All Loans	`list all loans`	N
List Incoming Loans	`list incoming loans`	N
List Outgoing Loans	`list outgoing loans`	N
Settle Incoming Loan	`paid [amount] /to [other party]`	Y
Settle Outgoing Loan	`received [amount] /from [other party]`	Y
Create A Bank Account Tracker	`bank-account [desc] /amt [initial amount of money] /at [initial date] /rate [interest rate]`	Y
Delete A Bank Account Tracker	`delete bank-account [index of the tracker]`	Y
Predict Balance	`check-balance [bank name] /at [the future date]`	N

Deposit	`Deposit [amount] [bank name] /at [date]`	N
Withdraw	`Withdraw[amount] [bank name] /at [date]`	N
List Bank Tracker	`list bank trackers`	N
Generate Current Month's Financial Report	`graph monthly report (histogram/line_graph/pie_chart)`	N
Generate the Graph of Expenditure/Income Categories	`graph income/expenditure trend (histogram/line_graph/pie_chart)`	N
Generate financial status	`graph finance status /until [date]`	N
Undo	`undo`	N
Timing Shortcuts	`now / ytd / tmr / lstwk / nxtwk / lstmth / nxtmth / lstyr / nxtyr` (replaces `d/m/yyyy`)	N/A
Exiting the program	`bye`	N