Moneytor

Use-Case Specification

Version 1.0

Revision History

| **Date** | **Version** | **Description** | **Author** |
| --- | --- | --- | --- |
| 12/11/2021 | 1.0 | Initial version | Team |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

**1.** [**Use-case Model**](#_gjdgxs) **4**

**2.** [**Use-case Specifications**](#_30j0zll) **5**

2.1. [Use-case: Track spending](#_1fob9te) 5

2.2. [Use-case: Share bill](#_3znysh7) 5

2.3. [Use-case: View spending](#_bq3linmzkkzu) 6

2.4. [Use-case: Maintain spending](#_srg7fpnj4qur) 6

2.5. [Use-case: Set spending goal](#_je1orppmpxo8) 7

2.6. [Use-case: Manage debt](#_rqh9w8e5arsx) 8

2.7. [Use-case: Set reminder](#_t1686x1daq62) 8

2.8. [Use-case: Remind spending note](#_g0tiica49lqz) 9

2.9. [Use-case: Note on homescreen widget](#_tcqv6rrnycwm) 9

2.10. [Use-case: Filter spending by category](#_m746nrl5n1sv) 10

2.11. [Use-case: Filter spending by time](#_i77ffvoevvse) 10

2.12. [Use-case: Analyze spending](#_eoj1i4miityu) 11

2.13. [Use-case: Visualize spending](#_i9hwrn8oznmi) 11

2.14. [Use-case: Log in with Google account](#_pvdzlp23gzzh) 12

2.15. [Use-case: Sync data](#_kt8v7vws2qzo) 12

2.16. [Use-case: Link to users’ E-wallet](#_oye1cnvczik3) 13

# Use-case Model



# Use-case Specifications

## Use-case: Track spending

| Use case Name | Track spending |
| --- | --- |
| Brief description | This use case describes how the User can add a new spending note to the local database |
| Actors | User |
| Basic Flow | 1. At the Main screen, the user clicks on ‘Add a spending’ button 2. System displays blank spending form 3. User enters the following information: spending title, category, cost, date, description 4. User clicks on ‘Done’ button 5. The system validates the data to insure the proper format 6. The system creates a new spending note |
| Alternative Flows | **Alternative flow 1: The information is invalid**   1. From step #5 of the basic flow, system displays an error message 2. Let the user try again from step #3   **Alternative flow 2: The user cancels enter spending note**   1. From step #4 of the basic flow, user clicks on ‘Cancel’ button 2. The system back to Main screen |
| Pre-conditions | User goes to Main screen |
| Post-conditions | The user successfully adds new spending note |

## Use-case: Share bill

| Use case Name | Share bill |
| --- | --- |
| Brief description | This use case allows the User to share the bill with other users, help detailed spending note |
| Actors | User |
| Basic Flow | 1. At the Track spending screen, the user clicks on ‘more detail’ 2. The system shows the blank ‘with who’ text field 3. User clicks on the ‘with who’ text field 4. The system brings up a form 5. User enters name, phone and cost of user who shared the bill and submit to the system 6. Steps 3-5 are repeated for each user shared the bill |
| Alternative Flows | **Alternative flow 1: User delete a shared user**   1. User clicks on ‘x’ button on name of that shared user   **Alternative flow 2: User edit information of a user shared this bill**   1. User clicks on name of that user 2. The system shows information of that user 3. User edit name, phone or cost of that user, and submit |
| Pre-conditions | User is in Track spending screen |
| Post-conditions | List of users who shared the bill is added to the spending note |

## Use-case: View spending

| Use case Name | View spending |
| --- | --- |
| Brief description | This use case allows the User to view list all of their spending notes |
| Actors | User |
| Basic Flow | 1. At the Main screen, the user clicks on ‘View spending’ button 2. The system retrieves and displays the list of spending note of the user |
| Alternative Flows | **Alternative flow 1: No spending note available**   1. The system displays the message “no spending note” |
| Pre-conditions | User goes to Main screen |
| Post-conditions | User can see list of all spending note |

## 

## Use-case: Maintain spending

| Use case Name | Maintain spending |
| --- | --- |
| Brief description | This use case allow the User to maintain spending note, this includes modifying and deleting spending from the system |
| Actors | User |
| Basic Flow | 1. At the View spending screen, the user selects a spending that he/she want to modify 2. The system displays full information of that spending 3. User clicks on ‘Edit’ button 4. The system enables editing 5. User edits information of that spending 6. User clicks on ‘Done’ button 7. The system validates the data to insure that the form is valid 8. The system updates that spending note |
| Alternative Flows | **Alternative flow 1: The information is invalidnote**   1. From step #7 of the basic flow, system display an error message 2. Continue step #5   **Alternative flow 2:** **The user cancels modify spending note**   1. From step #6 of the basic flow, user clicks on ‘Cancel’ button 2. The system backs to View spending screen   **Alternative flow 3: Delete a spending**   1. From #3 of the basic flow, user clicks on ‘Remove’ button 2. The system removes that spending   **Alternative flow 4: Swipe left or right to delete a spending**   1. At the View spending screen, the user swipe on a spending 2. The system removes selected spending from the database |
| Pre-conditions | User is in View spending screen |
| Post-conditions | User can delete and modify any spending note |

## Use-case: Set spending goal

| Use case Name | Set spending goal |
| --- | --- |
| Brief description | This use case allow user to set spending goal and notify when over spending |
| Actors | User |
| Basic Flow | 1. At the main screen, user click on “Set spending goal” button 2. System will display a list of spending goal you set before (sort by time descending), each item will have a button to delete or update 3. Click “Add” to add a new spending goal 4. A form appear for you to fill the information (spending goal and the time) and click “Done” to finish 5. If the information is valid, show “Success” popup, new spending goal will be added to database and display in the list at #2 |
| Alternative Flows | **Alternative flow 1: No spending goal added**   1. Display “No spending goal added”   **Alternative flow 2: Delete a spending goal**   1. From step #2 of basic flow, user clicks on an item they want to delete 2. Click delete icon 3. The system shows a popup to confirm, if the user confirms, delete that record.   **Alternative flow 3: Update a spending goal**   1. From step #2 of basic flow, user clicks on item they want to update 2. The system shows the form like step #4 of basic flow to edit the data and click save button. 3. If it’s invalid data, an error will display and you have to change it. If data is valid, update the database and display list, show “Success” message. If data is invalid, show error and ask the user to input again.   **Alternative flow 4: User overspending**   1. Notify user that they have overspended   **Alternative flow 5: Invalid data**   1. From step #4 of basic flow, the system shows an error message 2. Let the user input again from step #4 of basic flow |
| Pre-conditions | User is in Main screen |
| Post-conditions | User can see, add, update, delete any spending goal |

## 

## Use-case: Manage debt

| Use case Name | Manage debt |
| --- | --- |
| Brief description | This use case allow user to manage debt |
| Actors | User |
| Basic Flow | 1. At the main screen, the user clicks on the “Manage debt” button. 2. System will display a list of the user's debt or debt to the user (user can filter 2 types of debt with a filter button), each item will have a delete/update/contact button 3. Click floating (+) button to add new debt 4. A form appears for the user to input data (name, contact, debt, specific content) 5. User clicks on “Done” button to finish 6. The system validates the data to ensure the proper format. If data is valid, add a new record to the database, and display the list |
| Alternative Flows | **Alternative flow 1: No debt added**   1. Display “No debt added”.   **Alternative flow 2: Delete a debt**   1. From step #2 of basic flow, user clicks on an item they want to delete 2. Click delete icon 3. The system shows a popup to confirm, if the user confirms, delete that record.   **Alternative flow 3: Update a debt**   1. From step #2 of basic flow, user clicks on item they want to update 2. The system shows the form like step #4 of basic flow to edit the data and click save button. 3. If it’s invalid data, an error will display and you have to change it. If data is valid, update the database and display list, show “Success” message. If data is invalid, show error and ask the user to input again.   **Alternative flow 4: Contact**   1. From step # of basic flow, user clicks on ‘contact’ button on an item 2. The system opens that contact in the dialer |
| Pre-conditions | User is in manage debt screen |
| Post-conditions | Users can add, update, delete debt data, and contact the person for each debt. |

## 

## Use-case: Set reminder

| Use case Name | Set reminder |
| --- | --- |
| Brief description | This use case allows the user to set a fixed time to be reminded of spending notes. |
| Actors | User |
| Basic Flow | 1. The user opens the “Reminder” option in the Settings menu. 2. The system switches to the reminder screen with the “Reminder notification” option. 3. The user turns on the “Reminder notification” mode. 4. The system asks for notification permission. 5. The user clicks on the “Accept” button. 6. The system displays the “Set reminder” form below the “Reminder notification” option. 7. The user chooses the time of day they want to be reminded of spending notes and clicks on the “Save” button. 8. The system saves the created date of this reminder and the set time and reminder notification status is on. |
| Alternative Flows | **Alternative flow 1: “Reminder notification” status has been on**   1. The system switches to the reminder screen with the “Reminder notification” option and “Set reminder” form. 2. Continue #7 in the basic flow.   **Alternative flow 2: User doesn’t turn on “Reminder notification”**   1. From #3 of the basic flow, the user clicks on the “Back” button without turning on “Reminder notification” mode. 2. The system goes back to the Settings menu with nothing saved.   **Alternative flow 3: User doesn’t set reminder**   1. From #7 of the basic flow, the user goes back to the Settings menu. 2. The system uses the default reminder time and reminder type is daily.   **Alternative flow 4: User doesn’t accept notification permission**   1. From #5 of the basic flow, the user clicks on the “Cancel” button. 2. The system turns off the “Reminder notification” and continue step #2 in the basic flow. |
| Pre-conditions | The user is in the Settings menu. |
| Post-conditions | The user successfully turns on reminder notification and sets reminder time (day). |

## 

## Use-case: Remind spending note

| Use case Name | Remind spending note |
| --- | --- |
| Brief description | This use case allows the device periodically sends users notification reminding them to note down untracked spending in set time. |
| Actors | User |
| Basic Flow | 1. At the reminder time, the device displays a reminder notification on the home screen. 2. User clicks on this notification. 3. The system opens the tracking spending form on the app. |
| Alternative Flows | **Alternative flow 1: User ignores the notification**   1. User removes the reminder notification from the home screen. 2. The device ignores the notification. |
| Pre-conditions | The Reminder notification mode in the Daily reminder option of the settings menu has been turned on. |
| Post-conditions | User can open the tracking spending form on the app. |

## 

## Use-case: Note on home screen widget

| Use case Name | Note on home screen widget |
| --- | --- |
| Brief description | This use case allows the user to quickly note down their spending on a home screen widget |
| Actors | User |
| Basic Flow | 1. User clicks on the minimized Moneytor widget at home screen 2. System open the widget displays blank spending form 3. User enters the following information: spending title, category, cost, date, description 4. User clicks on ‘Done’ button 5. The system validates the data to insure the proper format 6. The system creates a new spending note |
| Alternative Flows | **Alternative flow 1: The information is invalid**   1. From #5 of the basic flow, system displays an error message 2. Continue step #3   **Alternative flow 2: The user cancels enter spending note**   1. From #4 of the basic flow, user clicks on ‘Cancel’ button 2. The system minimizes the widget and goes back to the home screen. 3. Continue step #1   **Alternative flow 3: The user removes the widget**   1. User drags the widget to the middle bottom of the home screen where it displays the bin icon 2. The system deletes the home screen widget and turns off “Display home screen widget” mode in the Settings menu |
| Pre-conditions | The minimized Moneytor widget is on the homescreen (The user agrees “Display home screen widget” when installing or setting the app). |
| Post-conditions | The user successfully adds new spending note |

## 

## Use-case: Filter spending by category

| Use case Name | Filter spending by category |
| --- | --- |
| Brief description | This use case allow user to view spending in selected categories |
| Actors | Users |
| Basic Flow | 1. At the Main screen, the user clicks on the ‘View spending’ button 2. The system retrieves and displays the list of spending notes of the user 3. User clicks on the ‘Filter by category’ button, then chooses the desired categories 4. The system filters and only shows spending data in the span of the duration chosen |
| Alternative Flows | **Alternative flow 1: No spending notes available**   1. Show “No spending notes available” regardless of the user’s filter choice |
| Pre-conditions | The user goes to the main screen |
| Post-conditions | The spending notes that belong to the chosen categories are shown |

## 

## Use-case: Filter spending by time

| Use case Name | Filter spending by time |
| --- | --- |
| Brief description | This use case allows the user to view spending filtered by date/time |
| Actors | Users |
| Basic Flow | 1. At the Main screen, the user clicks on the ‘View spending’ button 2. The system retrieves and displays the list of spending notes of the user 3. User clicks on the ‘Filter by time’ button, then chooses whether the application should show spending over last week/2 weeks/month 4. The system filters and only shows spending data in the span of the duration chosen |
| Alternative Flows | **Alternative flow 1: No spending notes available**   1. Show “No spending notes available” regardless of the user’s filter choice |
| Pre-conditions | The user goes to the main screen |
| Post-conditions | The spending notes with the noted date in the span of the chosen duration are shown |

## 

## Use-case: Analyze spending

| Use case Name | Analyze spending |
| --- | --- |
| Brief description | This use case allows the user to view detailed analysis of spending data, which can be filtered by time or category |
| Actors | Users |
| Basic Flow | 1. At the View spending screen, the user clicks on the ‘Analyze’ button 2. The system calculates and shows basic analysis such as average spending, highest/lowest spending… |
| Alternative Flows | **Alternative flow 1: No spending notes available**   1. Disable the ‘Analyze’ button upon the user going to the View spending screen   **Alternative flow 2: The user filters spendings before analyzing**   1. The system only shows analysis over the filtered data |
| Pre-conditions | The user goes to the View spending screen |
| Post-conditions | The basic analysis of the spendings is shown |

## 

## Use-case: Visualize spending

| Use case Name | Visualize spending |
| --- | --- |
| Brief description | This use case allows the user to view spending data visually by graphs |
| Actors | Users |
| Basic Flow | 1. At the View spending screen, the user clicks on the ‘Chart’ button 2. The system plots the data on a bar graph for chronological differences and a pie graph for categorical differences |
| Alternative Flows | **Alternative flow 1: No spending notes available**   1. Disable the ‘Chart’ button upon the user going to the View spending screen   **Alternative flow 2: The user filters spending notes by date/time first before visualizing**   1. Only plot the data satisfying the duration chosen by the user   **Alternative flow 3: The user filters spending notes by category first before visualizing**   1. Only plot the data which belongs to the chosen category by date and only show the bar graph if only one category is chosen |
| Pre-conditions | The user goes to the View spending screen |
| Post-conditions | The plotted graphs of the spendings are shown |

## 

## Use-case: Login with Google account

| Use case Name | Login with Google account |
| --- | --- |
| Brief description | This use case allow user to sign in to the app with as existing Google account |
| Actors | Users, Google Firebase |
| Basic Flow | 1. The user start the app for the first time 2. There are 2 buttons in the Login screen: “Continue with local account” and “Sign in with Google account”. If the user chooses to sign in with a Google account, a dialog will pop up and let the user choose an existing Google account on their phone or another one 3. The user get logged into the Main screen with their account |
| Alternative Flows | **Alternative flow 1: User is currently using a locally stored account**   1. The user go to account setting while using their local account 2. Press the button “Link with a google account”   **Alternative flow 2: User fail to log in a Google account**   1. Display error message 2. Bring user back to Login screen |
| Pre-conditions | The user has an existing Google account and internet connection. The user is at the login screen or in the user setting |
| Post-conditions | The user succeeds to login to the Main screen with all of their information match that of their Google account |

## 

## Use-case: Sync data

| Use case Name | Sync data |
| --- | --- |
| Brief description | Users that sign in with a google account will have all of their spending records uploaded to the cloud (Firebase). Also, a local account can also synchronize their current data when they link this account with a Google account |
| Actors | Users, Google Firebase |
| Basic Flow | 1. The user sign in with their google account 2. All of their data is automatically uploaded to the cloud |
| Alternative Flows | **Alternative flow 1: User is currently using a locally stored account**   1. The user go to account setting while using their local account and link it with a google account 2. The local account will have all of it data backed up to the cloud   **Alternative flow 2: The app is not connected to the internet for a while and the user has updated their spending records.**   1. When the internet is available, check if data is synchronized 2. Update the cloud data automatically |
| Pre-conditions | The user has an existing Google account and internet connection. The user is at the login screen or in the user setting. |
| Post-conditions | All user data is uploaded to cloud |

## 

## Use-case: Link to users’ E-wallet

| Use case Name | Link to users' E-wallet |
| --- | --- |
| Brief description | The user can link his account to an e-wallet so that all payment is automatically logged in to the database |
| Actors | Users, E-wallet API |
| Basic Flow | 1. User go to Setting -> Connect account to e-wallets 2. Choose an e-wallet service from the list 3. Authorise MoneyTor from the e-wallet app |
| Alternative Flows | **Alternative flow 1: E-wallet service decline to authorise our app + User login information is wrong**   1. Show error message and ask them to try again   **Alternative flow 2: The user has not linked to a Google account**   1. Show the link to Google account dialog 2. After the user succeed to do so, let him choose connect to a e-wallet service that he desire |
| Pre-conditions | User has internet connection, has e-wallet account and has linked to a Google account |
| Post-conditions | All user payments must be automatically logged into the database and updated to the cloud |