Budding Budget

owsenk	Kyle Owsen
lizs3434	Elizabeth Schibig
ischaaf	Isaac Schaaf
jbktsang	Jessica Tsang
hstefan	Stefan Holdener
mjsc	Maxton Scott Coulson

Use Cases and UI Diagrams

<u>Initial setup</u>

Preconditions: A student wants to be more financially aware and downloads our app on their phone.

Actor: User

Trigger: User opens app for the first time.

1. Upon opening the app for the first time, the system asks for the required setup information (assets, savings, recurring income, recurring charges, etc).
2. The user enters in all the necessary values.

   <u>Invalid Input</u>: Subcase of Initial Setup

   2.1 The user enters a value inconsistent with the requirements of the field.

   2.2 The system displays notification with a meaningful error message.

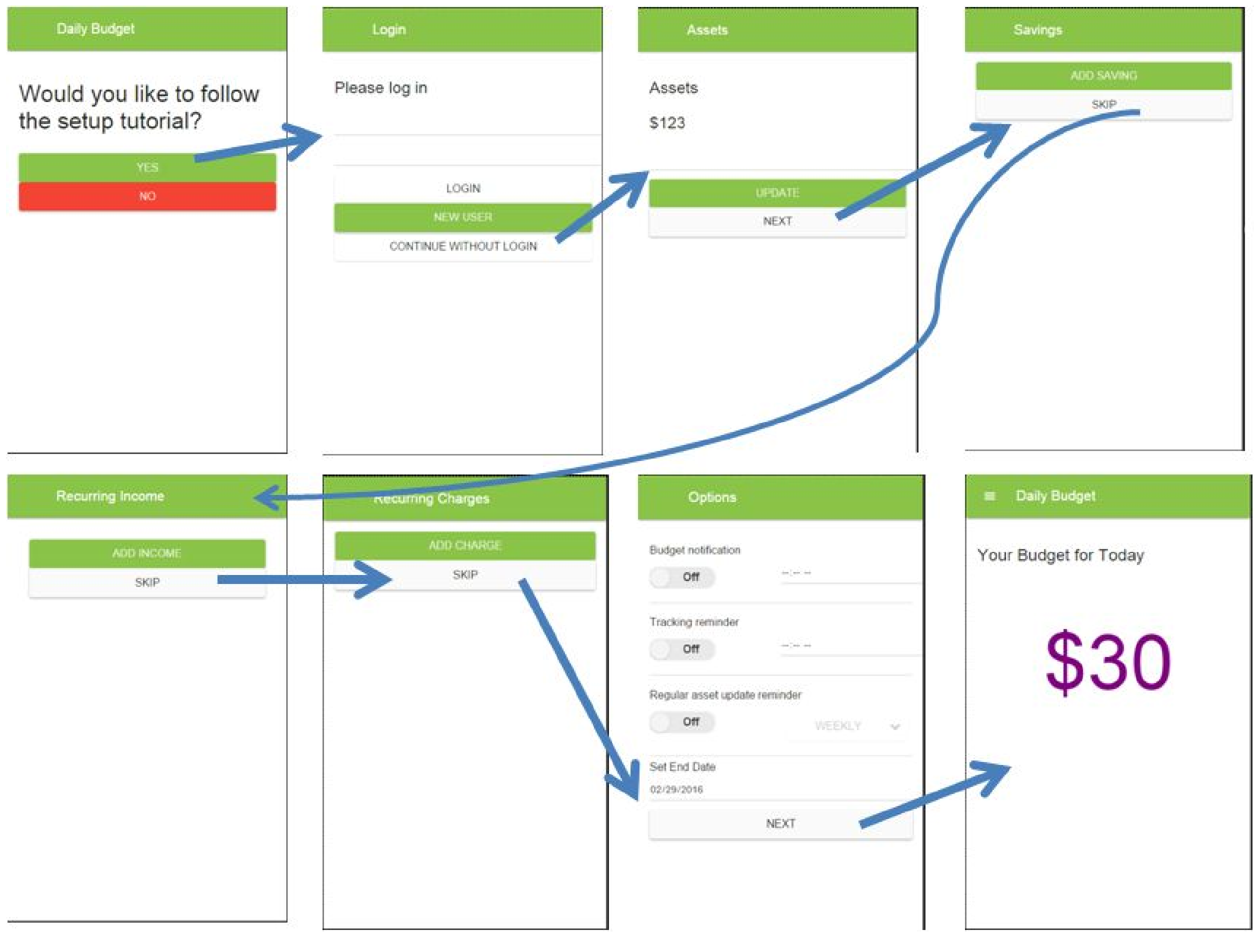
   2.3 Use case returns to step 2 of Initial Setup, fixing only the incorrect field.

3. The system gives option to <u>Setup Notifications.</u>
4. The setup saves/finishes and the user is taken to the main menu.

Success end condition: The input values are clear and the user can easily enter their information.

<u>Setup Notifications</u>: an optional feature

1. User chooses time/period that notifications are to occur.
2. The system saves preferences.


Would you like to follow the setup tutorial?

Please log in

Assets

$123

Budget notification Off --:-- --

Tracking reminder Off --:-- --

Regular asset update reminder Off WEEKLY

Set End Date 02/29/2016

Your Budget for Today

$30

<u>Update Daily Spending</u>:

Preconditions: Tracking notification enabled

Actors: User, System Clock

Trigger: System clock reaches the time specified in tracking notifications

1. The system provides phone notification that it's time to update daily spending.
2. User opens app (or it's already open).
3. User clicks on track spending button.
4. System displays track spending screen.
5. User inputs proper spending amount in textbox.
6. User presses UPDATE button.
7. The system saves spending amount.
8. The system recalculates budget plan.
9. Checks <u>Over/Under Spending.</u>
10. The system displays new tracked spending value.

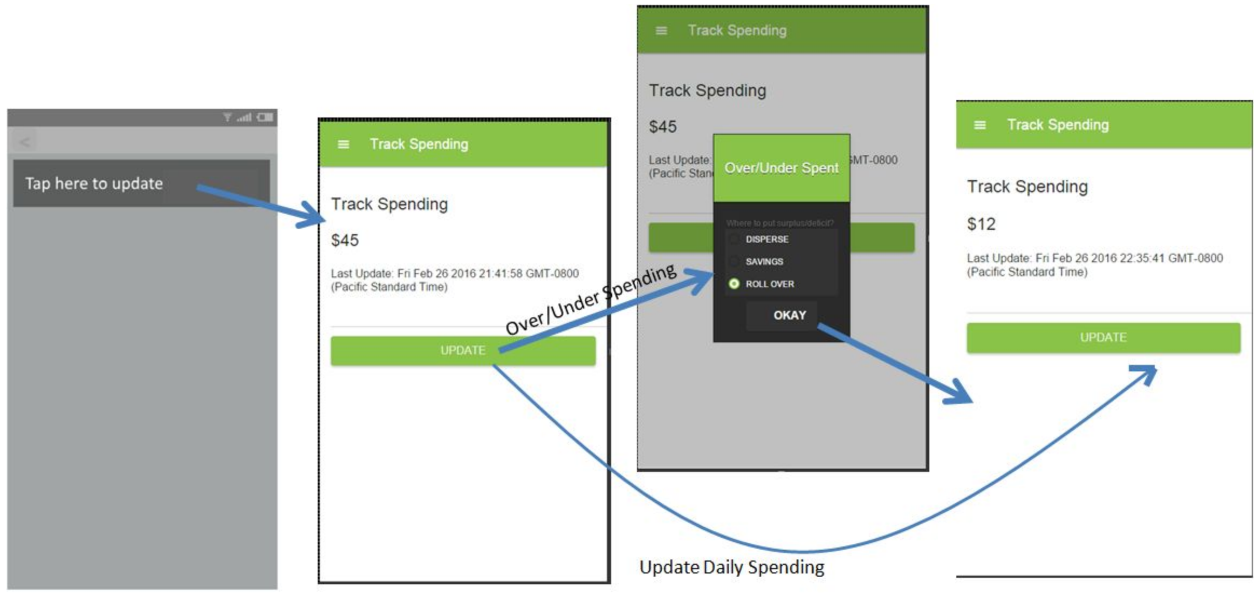
Success end condition: User easily updated spending and can view updated budget plan.

Alternate Case- <u>Over/Under Spending</u>:

Trigger: User over/under spends on their new budget plan

1. A pop-up gives the user options for dealing with the difference.
    a. Disperse: The user distributes the difference across the rest of the days.
    b. Savings: The user puts the difference in the savings (or another bucket).
    c. Roll-over: The user adds the entire difference to the next day.
2. The daily budgets following this day is adjusted correctly

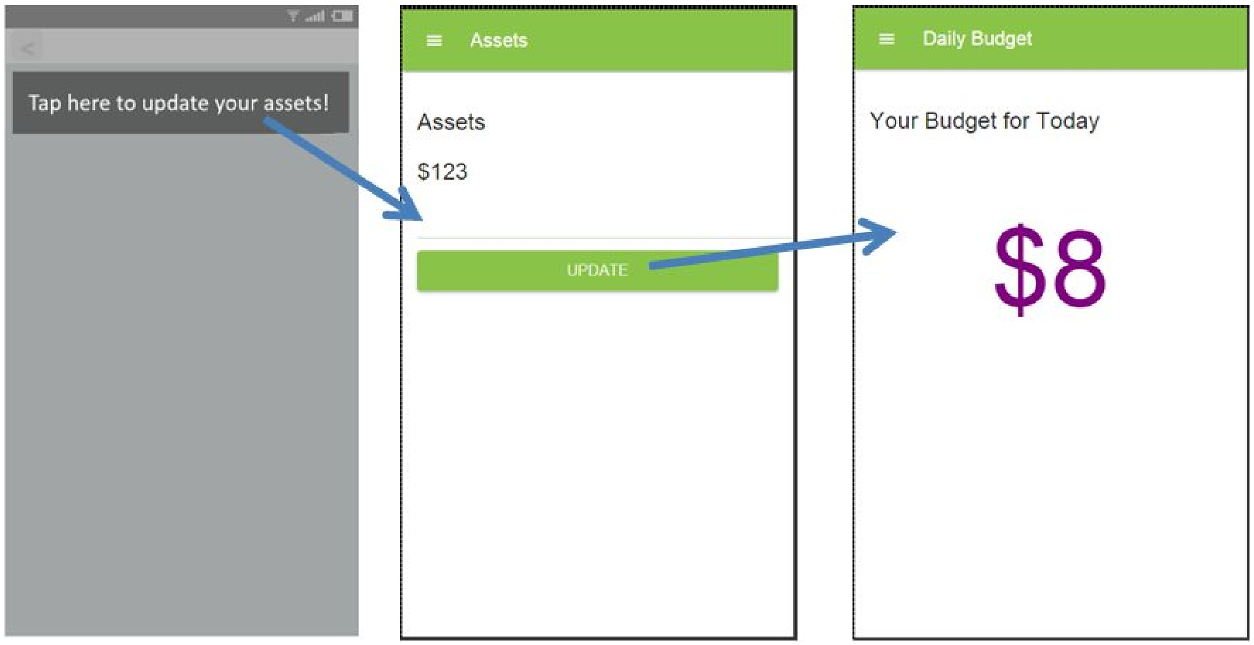
Success end condition: The spending is correctly adjusted for this day and the following days



Update Assets:
Preconditions: Tracking notifications enabled
Actors: User, System Clock
Trigger: System clock reaches the time specified to update assets

1. The system provides phone notification that it's time to check and update assets
2. User opens app (or it's already open).
3. User clicks on assets button.
4. System displays asserts screen.
5. User inputs proper assets amount in textbox.
6. The system saves spending amount.
7. The system recalculates budget plan.

Success end condition: User easily updated assets and can view updated budget plan.



These are the three most important use cases for our app. As stated in the product description, Budding Budget performs automatic calculation of daily budget and handling of over/under spending for a day. So it is critical to have these as use cases. The use case of initial setup is also major since everyone must pass this use case to use the app. And it's necessary to be able to update assets in order to make sure that the user doesn't get too out of sync with their actual asset value.