Flow 1: complete and ideal delivery

1. User 1 uploads a package (has enough money)
    a. taps package button in the middle
    b. fills in required details only
    c. taps request delivery
    - package is created with available status
    - reward of money is withdrawn
2. User 2 accepts that package (has enough money)
    a. taps on search tab
    b. filters by the same from/to parameters or weight
    c. taps on the package
    d. taps on accept delivery
    - is sent back to list
    - money_lock of money is withdrawn
    - package status is now accepted
    - package now appear under my deliveries
3. User 1 confirms pickup when the package is picked up
    a. taps on my packages
    b. taps on package
    c. taps confirm pickup
    - Is sent back to list
    - package status is now travelling
4. User 1 confirms delivery when package is delivered
    a. taps on my packages
    b. taps on package
    c. taps confirm delivery
    - Is sent back to list
    - money_lock + reward of money is now deposited to User 2
    - package status is now delivered

Flow 2: not enough money to upload package

1. User tries to upload a package (not enough money for reward)
    a. taps package button in the middle
    b. fills in required details only
    - request delivery button is disabled
    - the payment field should complain that the user has not enough money

Flow 3: not enough money to accept delivery

1. User tries to accept a package (not enough money for money_lock)
    a. taps on search tab
    b. filters and finds some package
    c. taps on the package
    d. taps on accept delivery
    - accept delivery button is disabled
    - the button should bring a toast that complains about insufficient balance

Flow 4: owner cancels delivery when available

1. User uploads a package (has enough money)
2. User cancels that package
    a. taps on the package
    b. taps on cancel delivery
    - is sent back to list
    - reward of money is deposited back to user
    - package is deleted

Flow 5: owner cancels delivery when accepted

1. User 1 uploads a package (has enough money)
2. User 2 accepts that package (has enough money)
3. User 1 cancels that package
    a. taps on the package
    b. taps on cancel delivery
    - is sent back to list
    - reward of money is deposited to owner
    - money_lock of money is deposited to assistant

Flow 6: assistant cancels delivery

1. User 1 uploads a package (has enough money)
2. User 2 accepts that package (has enough money)
3. User 2 cancels that package
    a. taps on the package
    b. taps on cancel delivery
    - is sent back to list
    - reward of money is deposited to owner
    - money_lock of money is deposited to assistant
    - package status is now available

Flow 7: complete delivery after owner cancel

1. User 1 uploads a package (has enough money)
2. User 2 accepts that package (has enough money)
3. User 1 cancels the order
4. User 3 accepts that package (has enough money)
5. User 1 confirms pickup when the package is picked up
6. User 1 confirms delivery when package is delivered

Flow 8: complete delivery after assistant cancel

1. User 1 uploads a package (has enough money)
2. User 2 accepts that delivery (has enough money)
3. User 2 cancels the delivery and package is listed as available again
4. User 3 accepts the delivery (has enough money)
5. User 1 confirms pickup when the package is picked up
6. User 1 confirms delivery when package is delivered

Flow 9: owner contacts assistant

1. User taps My Packages to see all current deliveries
2. User taps an accepted delivery and transitions to the detail view
3. User taps the link to the assistant profile and transitions to the assistants user profile, where a phone number and an email address should be visible.

Flow 10: assistant contacts owner

1. User taps My Deliveries to see all current deliveries
2. User taps any listed delivery and transitions to the detail view

3. User taps the link to the owner profile and transitions to the owners user profile, where a phone number and an email address should be visible.

Flow 11: user changes their profile info

1. A user in their own profile view taps on the Name button to change their name
    a. A popup with the title Name and their current name shows up
    b. User types a new name and clicks the OK button
    c. The popup goes away and their new name shows up on the Name button
    d. The change persists if the user transitions to another view and then transitions back to their user profile
2. A user in their own profile view taps on the Phone button to change their phone number
    a. A popup with the title Phone and their current number shows up
    b. User types a new phone number and clicks the OK button
    c. The popup goes away and their new number shows up on the Phone button
    d. The change persists if the user transitions to another view and then transitions back to their user profile
3. A user in their own profile view taps on the Mail button to change their mail address
    a. A popup with the title Mail and their current mail address shows up
    b. User types a new mail address and clicks the OK button
    c. The popup goes away and their new address shows up on the Mail button
    d. The change persists if the user transitions to another view and then transitions back to their user profile

Flow 12: register an account

1. User opens the Carrepsa app on their phone
2. User registers an account
    a. taps on sign up
    b. fills in details
    c. taps sign up
    - user is logged in

Flow 13: login to an account

1. User opens the Carrepsa app on their phone
2. User logins with that account
    a. taps on login
    b. fills in details

    c. taps on login
- user is logged in
- user is taken to the search tab

Flow 14: Update list to show newly added delivery requests

1. User opens the Carrepsa app on their phone
2. A new delivery is posted to Firebase
3. User pulls to refresh the search list
4. The new delivery request shows up in the list