Assumptions

* A tweet consists of:
  + name of the user that made the post
  + the screen name of that user
  + a message that informs how old the tweet is
  + the actual message that the user posted
  + the avatar image of the user
* App supports only portrait orientation
* App runs on iOS 7.1
* If the server connection is not receiving any data, but the call doesn’t fail the call is considered a success and the connection will not terminate
* If the connection fails 3 times, the user is presented with a error message with 2 options:
  + - * 1. Terminate the app
      * 2. Reset the counter and retry
* The tweets will be arranged in a table view and the first object entered is going to be the first one that is eliminated when the number of tweets exceeds 10. At this point for one second there will be 11 tweets on screen, so that we can eliminate the oldest tweet in an animated way
* The tweets will appear one by one until the maximum number is reached.
* if the tweet text is too long and it exceeds the height of 100 pts then the text will be scrollable and the textview that contains the text will not exceed 100 pts
* The user is responsible for connecting to the internet before starting the app
* The tableview cells user interaction is turned of since we do not need the user to be able to tap the cells.
* if the name, screen name, and posted … ago have invalid values (nil) the default values will remain on screen

Tests

The app was tested:

* Used Network Link Conditioner to simulate mobile internet connection
* tested with no internet connection
* tested with really long strings for the tweet message
* tested with invalid strings for tweet components

Test 1. Steps: Make sure the device is connected to the internet. Start the app.  
Expected: App should start and should begin receiving tweets one by one.

Test 2. Steps: Make sure the device is connected to the internet. Start app. Expected: App should populate the table view until it finished populating 11 cells, at that point the first cell (the oldest) should disappear in an animated way.

Test 3. Steps: Make sure the device is connected to the internet. Start app. Expected: Inspect every cell and verify if the all the elements are populated and fit the cell.

Test 4. Steps: Make sure the device is connected to the internet. Start app.   
Expected: The Network Activity Indicator should animate, indicating that the feed is working.

Test 5. Steps: Make sure the device is connected to the internet. Start app. Wait for a tweet with a very long message (more than 6 lines).  
Expected: The user should be able to scroll through the message and the message is not fully visible on screen.

Test 6. Steps: Make sure the device is not connected to the internet. Start the app  
Expected: The Network Activity Indicator should animate for a while and then an alert is presented to the user, with 2 options: Quit app and Retry.

Test 7. Steps: Make sure the device is not connected to the internet. Start the app  
Expected: An alert should be presented to the user with 2 options: Quit app and Retry.

Test 8. Steps: Make sure the device is not connected to the internet. Start the app  
Expected: The Network Activity Indicator should animate for a while and when the alert is presented to the user the Network Activity Indicator should not animate anymore.

Test 9. Steps: Make sure the device is not connected to the internet. Start the app and wait for the alert message and tap on the “Terminate app” option.  
Expected: The app should terminate and present the user with the device home screen.

Test 10. Steps: Make sure the device is not connected to the internet. Start the app and wait for the alert message.  
Expected: Check the console, you should find out that 3 attempts to connect were made.

Test 11. Steps: Make sure the device is not connected to the internet. Start the app and wait for the alert message and tap on the “Retry” option.  
Expected: The app attempts to connect to the API another 3 times then presents the user with the same alert message.