Assignment - iOS

Description

The product Tweeter allows users to post short messages limited to 50 characters each.

Sometimes, users get excited and write messages longer than 50 characters.

Instead of rejecting these messages, we would like to add a new feature that will split the message into parts and send multiple messages on the user's behalf, all of them meeting the 50 character requirement.

Example

Suppose the user wants to send the following message:

```
"I can't believe Tweeter now supports chunking my messages, so I
don't have to do it myself."
```

This is 91 characters excluding the surrounding quotes. When the user presses send, it will send the following messages:

```
"1/2 I can't believe Tweeter now supports chunking"
"2/2 my messages, so I don't have to do it myself."
```

Each message is now 49 characters, each within the allowed limit.

Requirements

1. **Create an iOS application that serves the Tweeter interface**. It will support the following functionality:
  a. Allow the user to input and send messages.
  b. Display the user's messages.
  c. If a user's input is less than or equal to 50 characters, post it as is.

d. If a user's input is greater than 50 characters, split it into chunks that each is less than or equal to 50 characters and post each chunk as a separate message.
e. Messages will only be split on whitespace. If the message contains a span of non-whitespace characters longer than 50 characters, display an error.
f. Split messages will have a "part indicator" appended to the beginning of each section. In the example above, the message was split into two chunks, so the part indicators read "1/2" and "2/2". Be aware that these count toward the character limit.

2. **The functionality that splits messages should be a standalone function**.
Given the above example, its function call would look like:

```
splitMessage("I can't believe Tweeter now supports chunking my
messages, so I don't have to do it myself.")
```

and it would return

```
["1/2 I can't believe Tweeter now supports chunking", "2/2 my
messages, so I don't have to do it myself."]
```

3. **The app must be written in Swift**.

4. **The business logic should be unit tested**. We have to know it works, right?

5. **The submission should be a git repository**. In the project directory, `git log` should show your commits.

6. Bonus points for any additional polish and sophistication you add to the experience.

Considerations - What are we looking for?

- We want to see how you create a new project and what technologies you decide to you use. A good project will be cleanly structured, will only contain the dependencies it needs, and will be well-documented and well-tested. What matters is not the technologies you use, but the reasons for your decisions. Bonus points will be given for demonstrating knowledge of modern Swift techniques and best practices.

- We want to see your command of core iOS technologies: Swift, Application and ViewController lifecycles, concurrency. The most critical part of this assignment is the message splitting functionality; so this should be in plain Swift without the use of libraries. The rest of the application may make use of utility libraries and frameworks.

- We want to see your sense of idiomatic iOS UI design. iOS is a visually rich and beautiful platform and we want you to display your sensibility in this area.

Submission

1. Please submit the project via email at tech-hiring at zalora dot com, or via a Github link.
2. You have one week to complete the assignment. Please write in if you have special circumstances where you cannot complete it on time.