MS Project Report

E-cigarette Survey Prompt Application for Android Mobile Devices

Author: Galen K. Thomas-Ramos

UIN: 669242905

College of Engineering

Department of Computer Science

University of Illinois-Chicago

Semester of Graduation: Summer 2014

Advisor: Dr. Andrew Johnson

Secondary Committee Member: Dr. Robert Kenyon

Table of Contents Page

1. Abstract

The e-cigarette survey prompt application is designed for Android smartphones. This mobile application allows researchers at the University of Illinois at Chicago to gain self-regulated data from all users involved in the study. This report details a previous project application, from which the application was design. A detailed explanation of the capabilities and user interactions with the application is documented through figures and text. Finally, the changes and complications which arose during the application's development are discussed, with the final product outlined at the end of the report.

2. Introduction

2.a. Importance of Project

The effects of tobacco use have been long studied and are well understood. The carcinogens from smoking tobacco have been associated with a wide swath of terminal or debilitating ailments. Because of the serious health risks associated with smoking tobacco, cigarette use is well-regulated with long-standing legislation in place to help reduce its impact on individuals and the public.

Recently, E-cigarettes have been developed and advertized as an acceptable, healthier alternative to smoking traditional tobacco cigarettes. However, there is no regulation in place for the use of e-cigarettes. E-cigarettes and electronic vaporizers are fairly new to the market and as of yet there are no warning labels on electronic tobacco paraphernalia, nor is there a ban on the sale of E-cigarettes to minors. [1, 2] In order for the Food and Drug Administration to regulate the manufacturing, marketing, and consumption of e-cigarette products, data must first be gathered on the characteristics of its users.

Some of the main users of electronic tobacco products are young adults and people who want to quit smoking. [3, 1, 4] A pervasive public opinion of e-cigarette users is that the vapor which they produce is not as harmful as the carcinogenic tobacco smoke of regular cigarettes. [3]Unfortunately, not enough research reports have been published to substantiate this belief. [5, 2]The aim of this project was to better understand the usage patterns of young e-cigarette users and supply the federal Food and Drug Administration with these data.

2.b. Previous Work

Previous research using Personal Data Assistants (PDAs) to garner connections between adolescent smokers and their environment has been long standing and well-practiced [6, 7]. The ability of PDAs to gather real-time data from its users has provided researchers with a powerful tool in time-sensitive data collection [8]. One drawback to this approach, however, is the necessity of having to carry an additional device. Because of this, development of a similar application using which utilizes one's own phone was desired.

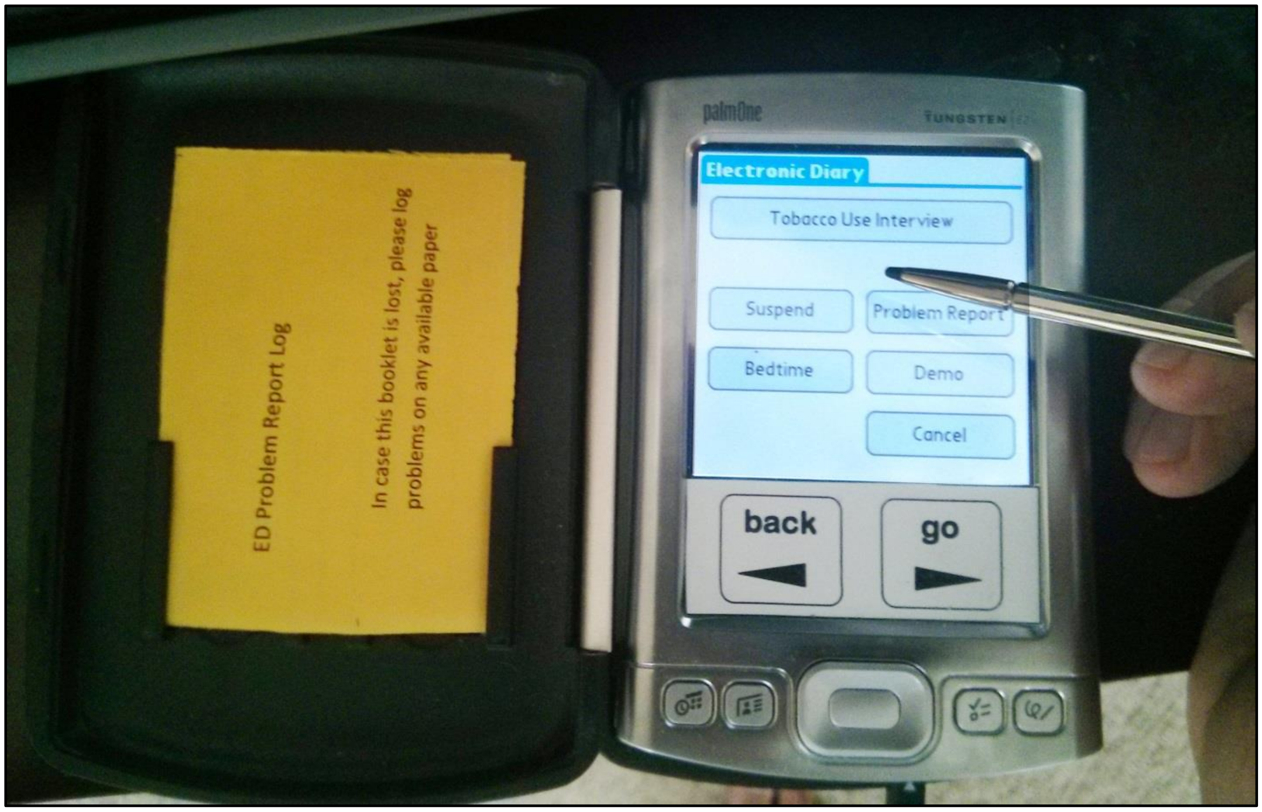


Figure 1: Photograph of the *PalmPilot* device running the previous tobacco-use survey application.

The precursor to the Android smartphone application outlined in this report was developed for a *PalmOne PalmPilot* (Figure 1). The project was funded by the US Food and Drug Administration and described as an Ecological Momentary Assessment. [9] Three core components were outlined for the development of the application: random prompt notification capabilities, suspension of notification capabilities, and two types of tobacco event reporting: random prompt event reporting and user-initiated event reporting. In addition to these core functions, the previous application boasts the ability to turn off notifications while a user sleeps with its 'Bedtime' menu option. A user could also report a problem with the application and take an example tobacco use survey from the main menu by accessing the 'Demo' feature (Appendix F.1).

Application development using the Palm Pilot platform allowed the research team a great amount of freedom and control over the device's operation. Users could not diminish the volume of the device and were therefore apt to answer more notifications. Also, researchers were able to restrict the control of power to the *PalmPilot*: once the device was turned on, and the application started, users could neither turn off the device by pressing the power button nor could they access other areas of the Device. Those involved in the user study were constrained to using the device for the application alone. This was enforced not only in the application—there was no option within the application to go back to the *PalmPilot*'s main menu—but also in the hardware of the device, as user interaction with the buttons at the bottom of the device was disabled. Such control over the device proved to be a great benefit to data collection.

3. Project Outline

3.a. Needs of the Research Group

At the onset of the project, the research group spent the first two months outlining the needs of the application. The core functionality of the smartphone application had to mirror that of its *PalmPilot* progenitor, i.e., the application had to support random prompt notifications, the suspension of those notifications, and incorporate two separate tobacco use interviews: random prompt interviews and user-initiated interviews. Aside from the above, core functionality, several additional functions were desired by the research group: putting the device to sleep, encrypting and hiding all user data, and delaying any current notification within the application.

From the first meeting, building a cross-platform application which would work on both Android and iOS was lauded as a chief desire. Because of this, the group decided to pursue application development using a web-based approach and settled on using *PhoneGap* in the creation of the application because of its advertised cross-platform development capabilities.

3.b. Development Package

In using *PhoneGap* as the development platform, the research group was committing to a hybrid avenue of software engineering (Figure 2). The method for developing *PhoneGap* applications required the use of several different programming languages and Application Programming Interfaces (API), as well as an intimate knowledge of WebView execution priorities within a given Mobile Operating System (OS). *PhoneGap* relies heavily on open source, community maintained plug-ins for certain extensible features not supported by *PhoneGap*.

PhoneGap utilizes several different languages during development, including Android, Javascript, HTML and CSS. An intimate knowledge of asynchronous code execution is vital to the desired application performance. Also, having a firm understanding of the lowest levels of mobile OS operations in relation to maintaining a WebView application proved paramount to the application's reliable and intended operation.

Below are the architectural components of the E-cigarette Survey Prompt Application:

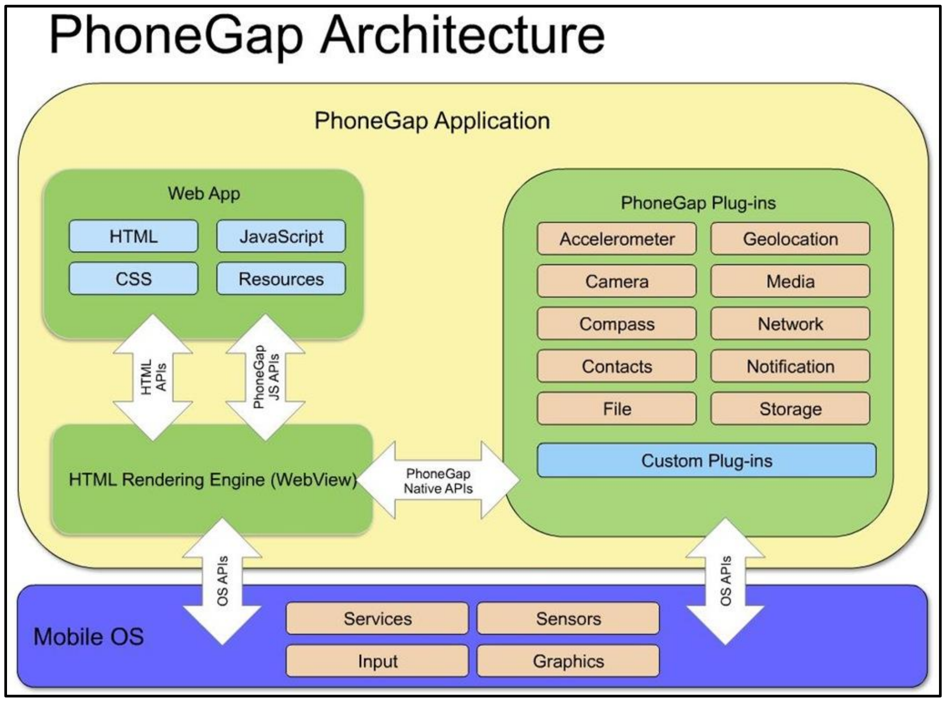


Figure 2: Diagram of the *PhoneGap* application and Mobile Operating System interface. Counter-clockwise from the Web App container: The **Web App** contains all of the web based code within the *PhoneGap* application—this is where the

majority of the application's code can be found. Through HTML and *PhoneGap* APIs, the Web App communicates with the **HTML rendering engine** to display the application as a **WebView**—exactly like any browser application displays web page contents. ; The **Mobile OS** provides access to its services and sensors through the operating systems APIs. ; Similarly, the Mobile OS also communicates with ***PhoneGap* plug-ins**—through the device's native language—in order to extend *PhoneGap* functionality. ; Finally, the ***PhoneGap* plug-ins** are able to communicate and reflect changes in the WebView through *PhoneGap*'s Native APIs.

3.b.i Android SDK

The Android Software Development Kit (SDK) provides the necessary tools to create applications for Android devices. The API used in this project was API 16, the repository developed for use with Android 4.1.2 devices. Direct development in Android was seldom necessary during this project because the *PhoneGap* API bridged communication between the mobile OS and the web application.

3.b.ii *PhoneGap*

According to the *PhoneGap* website, "*PhoneGap* is an application container technology that allows you to create natively-installed applications for mobile devices using HTML, CSS, and Javascript." [10]Regarding the *PhoneGap* API, "…enables you to access native operating system functionality using Javascript. You build your application logic using Javascript, and the *PhoneGap* API handles communication with the native operating system." [10] This means that there are two separate APIs that constitute *PhoneGap:* one Javascript API that provides an interface for the developer, and another API that is written in a language native to the mobile OS of the device. This second API is rarely interacted with, save for the development of *PhoneGap plug-ins*.

One of the most important aspects of *PhoneGap* is its ability to host user-made plug-ins. This allows the open-source *PhoneGap* community to extend the use of *PhoneGap* by enabling native features which are not accessible with the core package. Developing *PhoneGap* plug-ins does require platform-specific knowledge because the plug-ins communicate directly with the mobile OS using the device's native language.

3.b.iii jQuery Mobile

jQuery Mobile is advertised as, "an HTML5-based user interface system designed to make responsive web sites and apps that are accessible on all smartphone, tablet and desktop devices". [11]This Javascript and CSS library is dependent on an additional Javascript library, *jQuery* [12], which expands the use of Javascript by diversifying selectors, event handling, and Ajax manipulation. *jQuery Mobile* is a UI library which utilizes *jQuery* to implement robust widgets and widget use within any mobile web application.

3.c Hardware

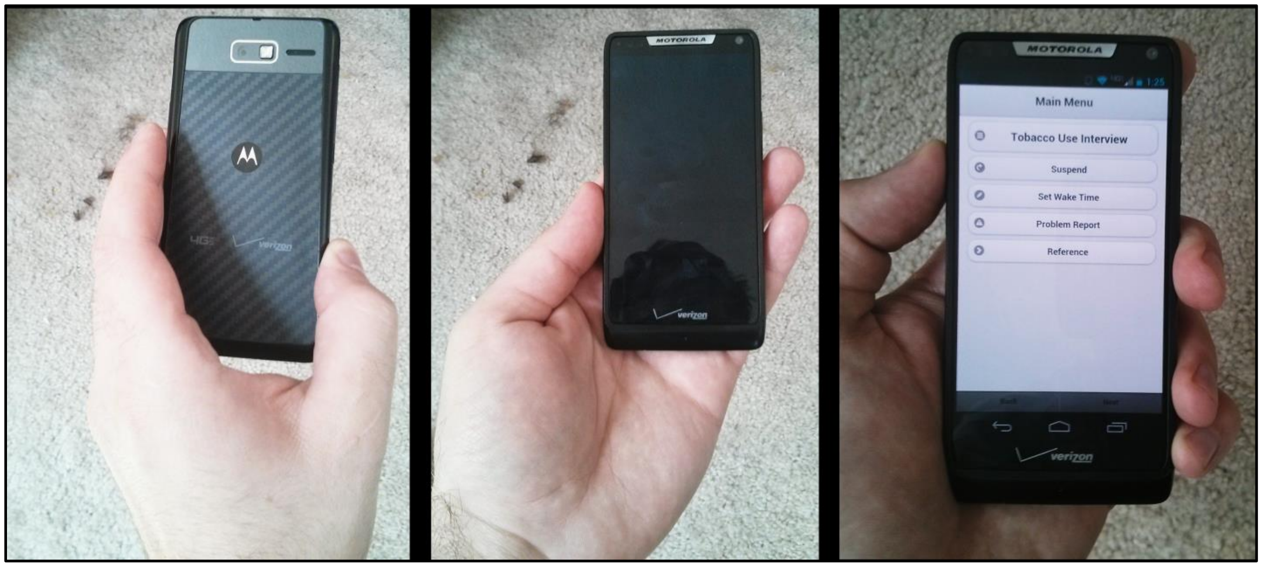
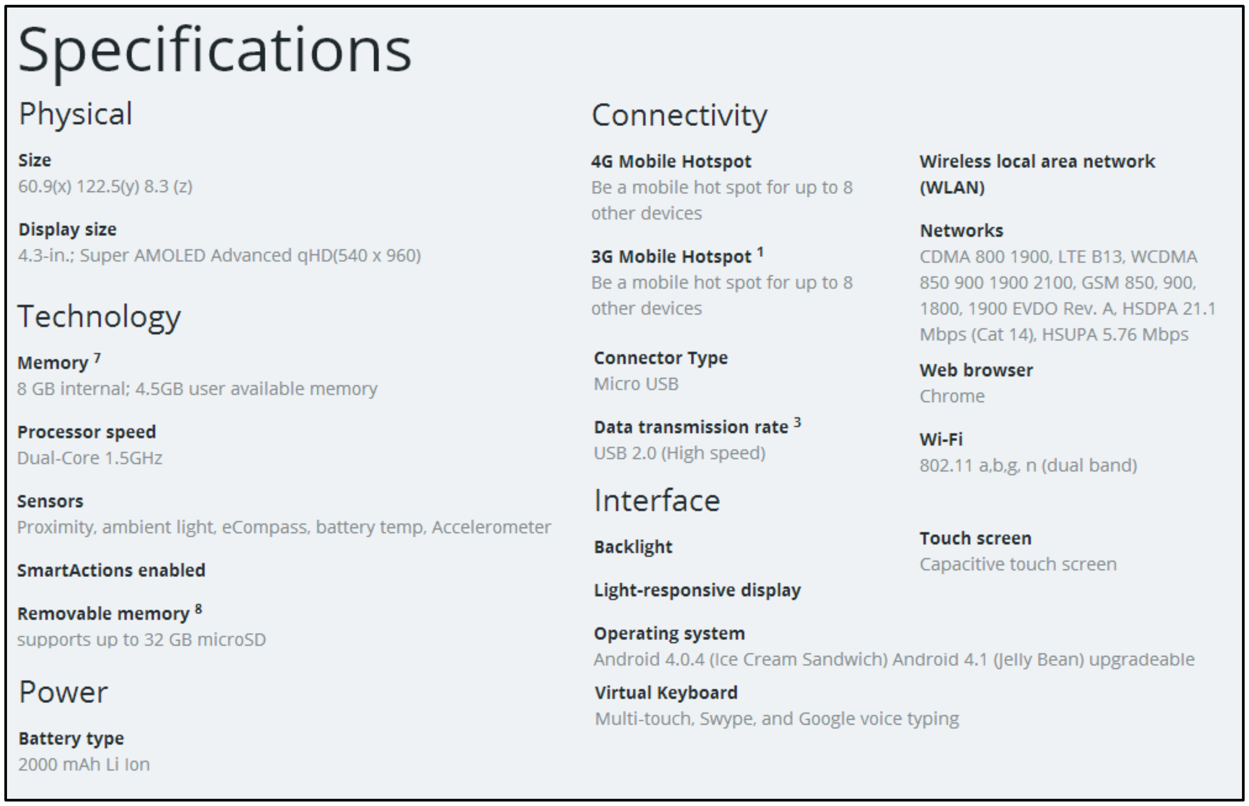


Figure 3: Photographs of the Android device, a *Motorola Razr M*. From left to right: the back and front of the smartphone followed by the main menu of the e-cigarette application running on the device.

The Android smartphone device leased by the research group for this project was a *Motorola Razr M*. [13] This phone used Android 4.1.2 –or Jellybean—as its active operating system. Relevant hardware specifications for this phone can be found below (Figure 4).


Specifications

Physical

Size
60.9(x) 122.5(y) 8.3 (z)

Display size
4.3-in.; Super AMOLED Advanced qHD(540 x 960)

Technology

Memory [7]
8 GB internal; 4.5GB user available memory

Processor speed
Dual-Core 1.5GHz

Sensors
Proximity, ambient light, eCompass, battery temp, Accelerometer

SmartActions enabled

Removable memory [8]
supports up to 32 GB microSD

Power

Battery type
2000 mAh Li Ion

Connectivity

4G Mobile Hotspot
Be a mobile hot spot for up to 8 other devices

3G Mobile Hotspot [1]
Be a mobile hot spot for up to 8 other devices

Connector Type
Micro USB

Data transmission rate [3]
USB 2.0 (High speed)

Wireless local area network (WLAN)

Networks
CDMA 800 1900, LTE B13, WCDMA 850 900 1900 2100, GSM 850, 900, 1800, 1900 EVDO Rev. A, HSDPA 21.1 Mbps (Cat 14), HSUPA 5.76 Mbps

Web browser
Chrome

Wi-Fi
802.11 a,b,g, n (dual band)

Interface

Backlight

Light-responsive display

Touch screen
Capacitive touch screen

Operating system
Android 4.0.4 (Ice Cream Sandwich) Android 4.1 (Jelly Bean) upgradeable

Virtual Keyboard
Multi-touch, Swype, and Google voice typing

Figure 4: Specifications for the *Motorola Razr M* used in the development of the e-cigarette survey prompt application. [13]

Data loss due to unreliable network connectivity was a chief concern of the research group, specifically, in regards to meeting the Institutional Review Board's project approval. Therefore, the application was developed without network connectivity in mind. Each phone has access to 3G and 4G networks, as well as Wi-Fi networks, but their availability had no influence on the application's performance.

3.d. Application Outline

3.d.i. Registration and Login

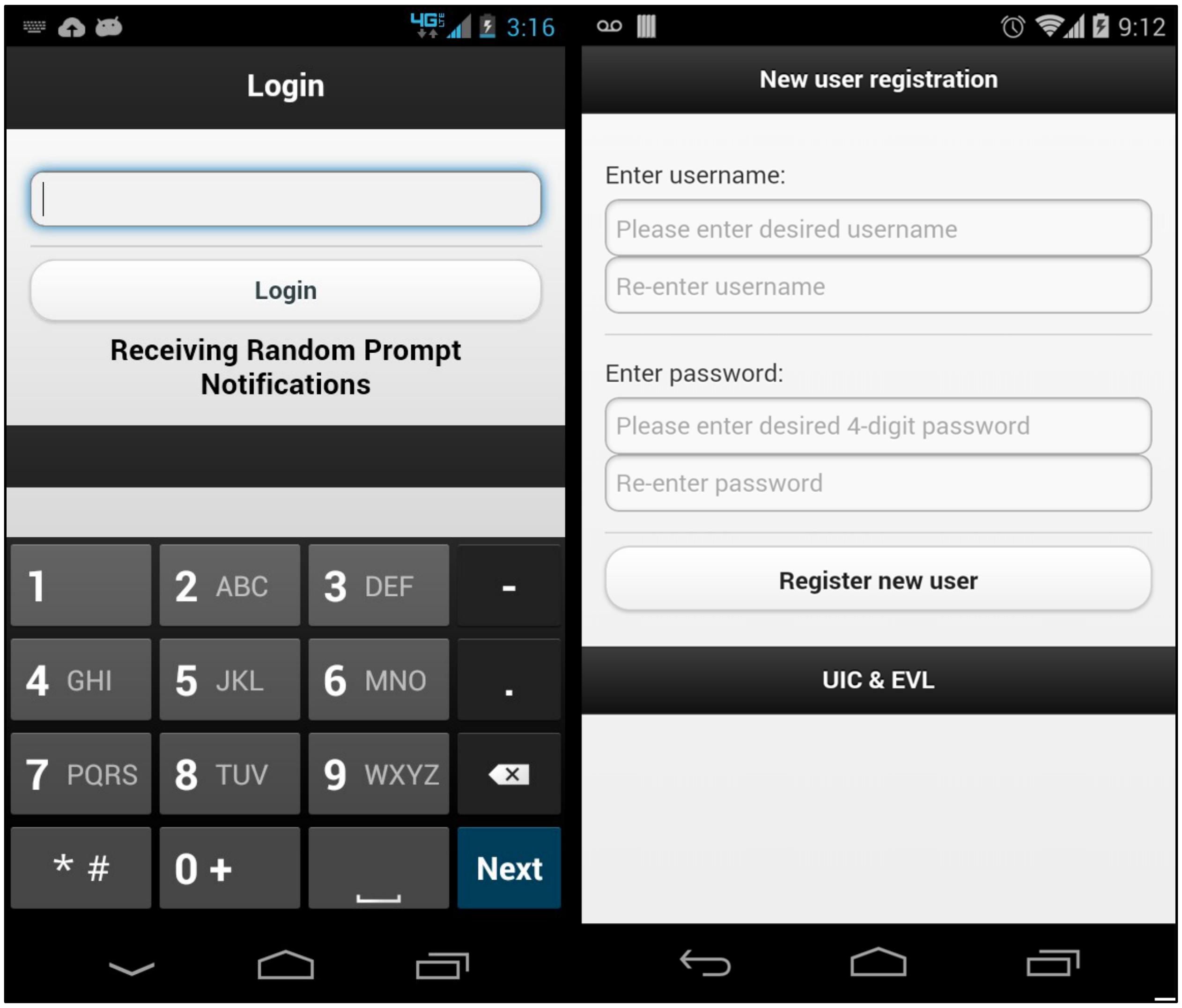


Figure 5: Screenshot composition of login pages within the running e-cigarette survey application. From left to right: the initial login screen for the application; the screen for new user registration.

The login and registration pages of the e-cigarette survey prompt application (Figure 5) had the following features:

- The login screen changed depending on whether or not the user has already registered and had a username and password saved to the device.
- If the user was new and did not have login credentials, a new user registration screen appeared asking the user to register their information (Figure 5).
    - A user must enter a username, with no restrictions to the name itself.

    - A user must also enter a 4-digit pin number used during subsequent logins. If a pin was entered incorrectly the process started again and the application asked the user to re-enter their credentials.
    - A user could not access the main portion of the application without having registered as a new user.
- Once a user registered their login credentials, a different login screen allowed the user to enter the application (Figure 5).
- The only time that a user could revisit the new user registration screen was if a member of the research group deletes their credential data.
- User credential data were saved to a 'log.log' file which held a single line of text in the form of 'userID, xxxx'; where xxxx was the user's 4-digit pin. A comma was used as the active delimiter.
- The presence of the 'log.log' file determined whether the login screen defaulted to the new user registration screen or the user login screen.
- If a user entered the incorrect pin number at the user login screen, they were notified of the error and asked to try again.
- The only way for a user to access the main portion of the application was to correctly enter the registered pin number.

3.d.ii. Interviews

During the preliminary study of this project, eight young adults were given a Motorola Razr M Android smartphone pre-loaded with the e-cigarette survey application. These users were of an ideal set of students as they were involved in the previous PDA study mentioned above. They were to have the smartphone available to answer survey questions during all waking hours. Data on e-cigarette usage were collected using two types of surveys. One survey (the Tobacco Use Interview) was to be completed after each usage of a tobacco product. For the second survey (the Random Prompt Interview) the user was randomly alerted by the smartphone through a loud notification that the user should log onto the application to take a survey.

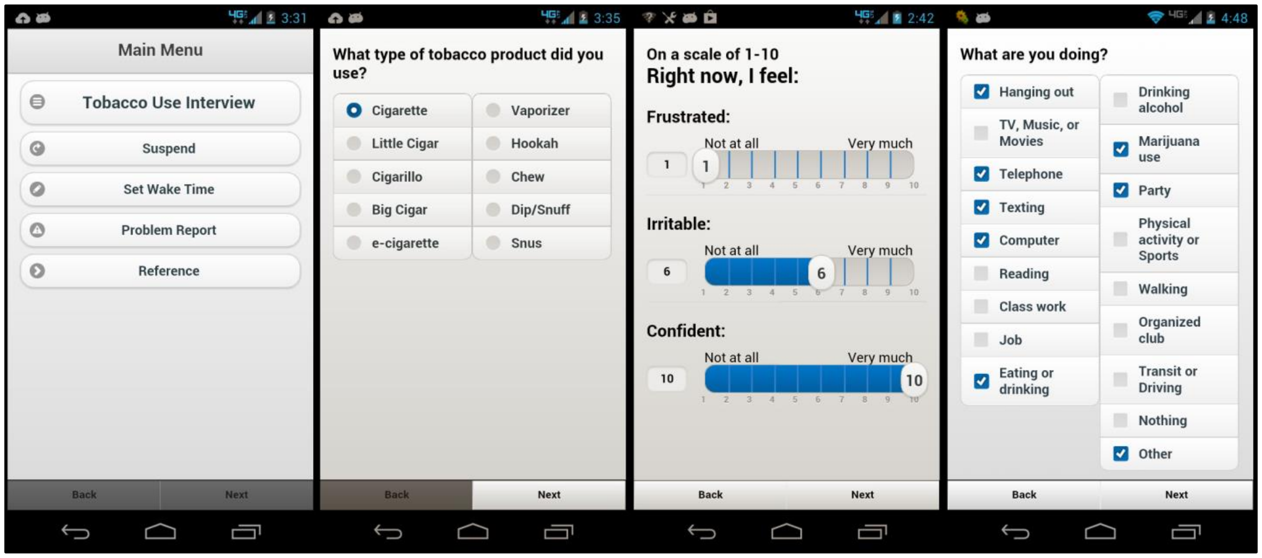
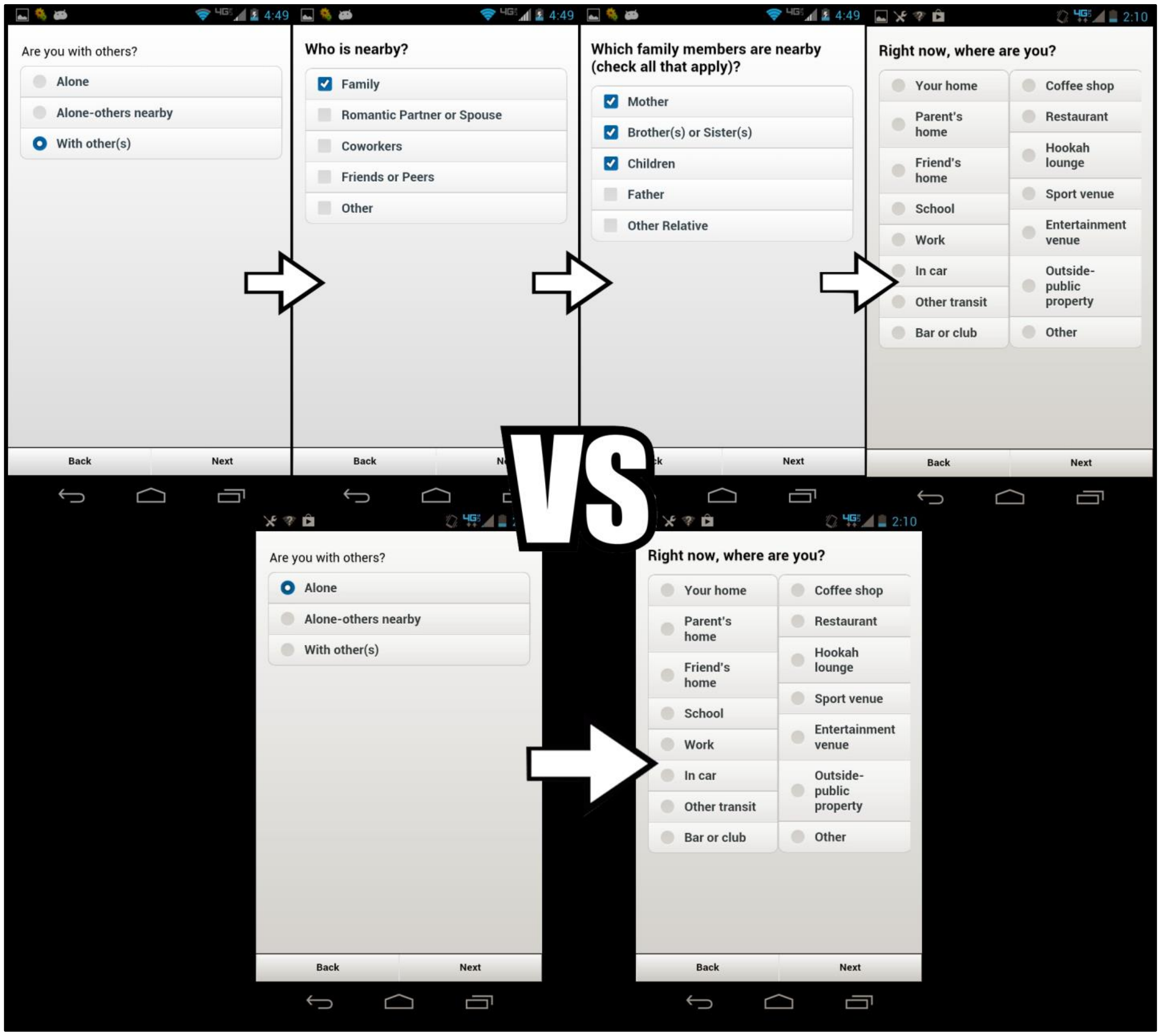


Figure 6: Screenshot composition of the different widget types within the running e-cigarette survey application. From left to right: the main menu of the e-cigarette survey prompt application; radio widget group; customized slider widget group; and checkbox widget group.

3.d.iii Tobacco Use Interviews

- Tobacco Use Interviews were surveys to be completed by the user on the smartphone after finishing a tobacco product.
- This set of questionnaires was different than the Random Prompt Interview survey (Appendix B).
- The survey was initiated by logging into the application and pressing the button labelled 'Tobacco Use Interview' from the main menu (Figure 6).
- The user was guided through a series of questions which utilized checkboxes, radio boxes, and sliders to record their responses. If a user did not answer all of the questions within a given page of the survey, they were not allowed to continue to the next page.



Are you with others?
- Alone
- Alone-others nearby
- [✓ With other(s)]

Who is nearby?
- [✓ Family]
- Romantic Partner or Spouse
- Coworkers
- Friends or Peers
- Other

Which family members are nearby (check all that apply)?
- [✓ Mother]
- [✓ Brother(s) or Sister(s)]
- [✓ Children]
- Father
- Other Relative

Right now, where are you?
- Your home
- Parent's home
- Friend's home
- School
- Work
- In car
- Other transit
- Bar or club
- Coffee shop
- Restaurant
- Hookah lounge
- Sport venue
- Entertainment venue
- Outside-public property
- Other

Are you with others?
- [✓ Alone]
- Alone-others nearby
- With other(s)

Right now, where are you?
- Your home
- Parent's home
- Friend's home
- School
- Work
- In car
- Other transit
- Bar or club
- Coffee shop
- Restaurant
- Hookah lounge
- Sport venue
- Entertainment venue
- Outside-public property
- Other

Figure 7: Screenshot composition of the running e-cigarette survey application. Two possible interview questionnaire routes; the top results in additional pages of survey questions, while the bottom does not.

- Depending on their answers, users might have to fill-out one or two additional questions within the survey (Figure 7).
- The survey input was only recorded to the .tob file (the file which keeps a record of all of the user's Tobacco Use Interview answers) once they answer ALL questions within the survey.
- If a user leaves a Tobacco Use Interview survey, the application will NOT record the survey as having been abandoned or missed (Discussed below). This mechanic only applies to Random Prompt Interview surveys.
- If a user logs in to the application within three minutes of a notification firing they will be taken to the starting questionnaire of a Random Prompt Interview, and not be allowed to access the Tobacco Use Interview from the main menu.

3.d.iv Random Prompt Interviews

- Random Prompt Interviews were surveys to be completed by the user within three minutes of a Random Prompt Notification firing.
- This set of questionnaires was different than the Tobacco Use Interview survey (Appendix C).
- This survey could be initiated by logging into the application within three minutes of a notification firing.
- If there were multiple notifications within Android's notification area, the user would only be able to initiate the Random Prompt Interview based off of the most recently fired notification. All other notifications were recorded as having been missed (Discussed below).
- The user was brought through a series of questions which utilized checkboxes, radio boxes, and sliders to record their responses.
    - If a user did not answer all of the questions within a given page of the survey, he/she was not allowed to continue to the next page.
- Depending on the answers, the user might have been requested to fill out one or two additional questions within the Random Prompt Interview survey (Figure 7).
- The survey input was only recorded to the '.rpt' file (the file which kept record of all of the user's Random Prompt Interview answers) once they answer ALL questions within the survey.
- If a user wanted access to the main menu of the application within three minutes of the last notification firing, he/she first had to take a Random Prompt Interview survey.
    - If a user began, but did not finish a Random Prompt Interview survey by exiting out of the application in the middle of the Random Prompt Interview survey, the survey was recorded as having been abandoned along with a timestamp at the point of abandonment (Discussed below).
    - The user did NOT receive an alert that the survey has been recorded as abandoned.
- If a user began a Random Prompt Interview Survey but did not finish it within five minutes, that survey was recorded as having been abandoned (Discussed below).
    - This could only occur if the display setting of the Android device was set to remain open for at least five minutes.
    - The user DID receive an alert that they had taken too long to complete the survey and had been recorded as abandoned.

3d.v Time Functions

Two functions within the application were developed for the convenience of the user. These functions allowed the user to cater local notifications around their own schedule. One such function suspended future notifications for a selected short period of time. The other function allowed a user to put the application to sleep for a longer period of time, during which time they did not receive any notifications. An alarm signaled the user that they had begun to receive notifications again.

3.d.vi Suspending Notifications

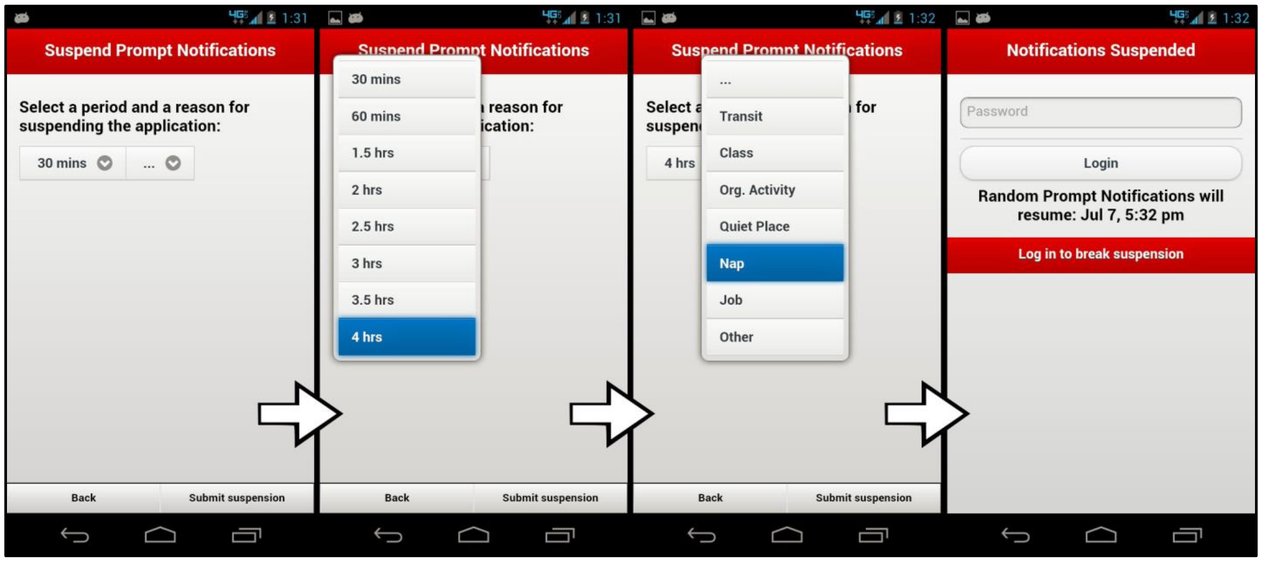
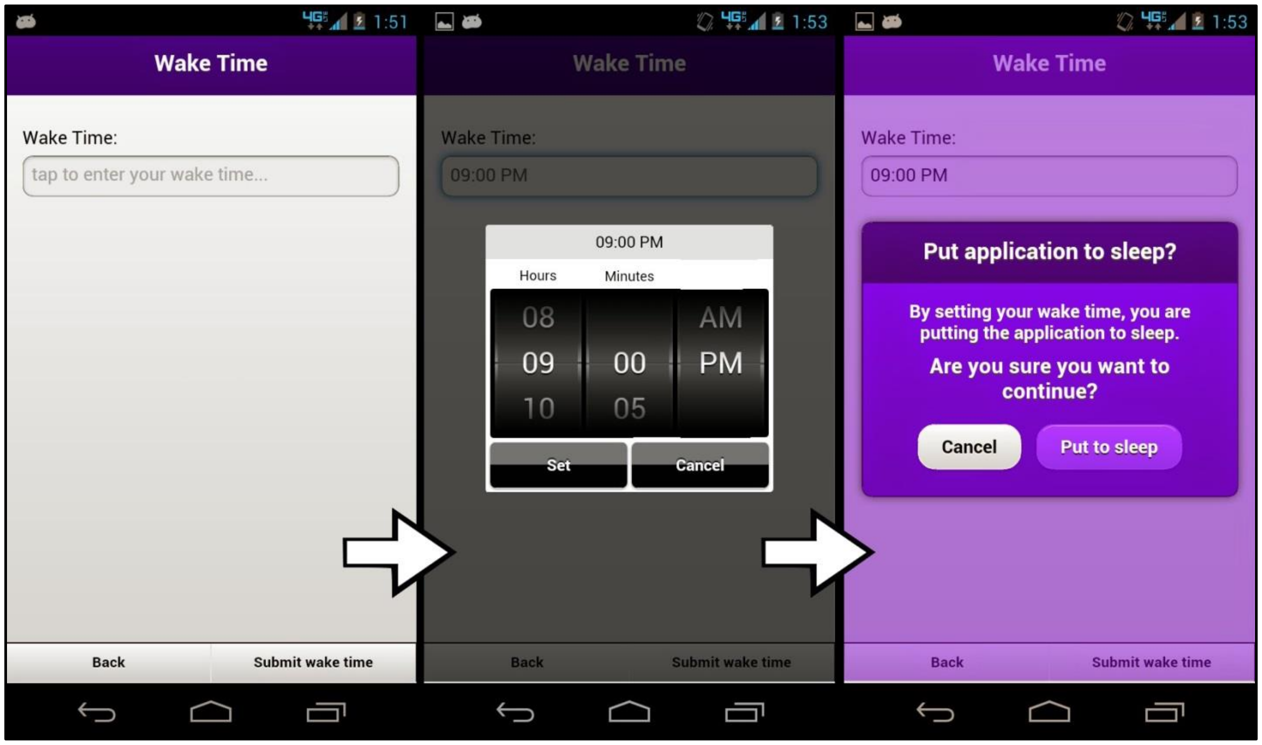

Suspend Prompt Notifications
Select a period and a reason for suspending the application:
30 mins
...
Back
Submit suspension
30 mins
60 mins
1.5 hrs
2 hrs
2.5 hrs
3 hrs
3.5 hrs
4 hrs
...
Transit
Class
Org. Activity
Quiet Place
Nap
Job
Other
Notifications Suspended
Password
Login
Random Prompt Notifications will resume: Jul 7, 5:32 pm
Log in to break suspension

Figure 8: Screenshot composition of the the running e-cigarette survey application's suspension of prompt notifications. From left to right: initial Suspension Prompt Notification page; time options drop down widget; reason for suspension drop down widget; resultant login screen after suspending notifications.

- A user could access the suspension function from the main menu of the application by tapping on the 'Suspend' button.
- The user was taken to the 'Suspend Prompt Notifications' page where he/she could change options related to the suspension (Figure 8).
  - A user had to choose an amount of time for which they wanted to suspend notifications.
  - A user must also choose a reason for the suspension.
    - If a user did not enter both an amount of time for suspension and a reason for suspension, he/she was alerted and not allowed to suspend any notifications.
    - This information was not recorded because the research group did not regard it as important.

- Once a user completed the steps to suspend notifications, they were brought back to the login page of the application (Figure 8).
    - The login page now conveyed the suspended status of the application with a red themed border profile.
    - There also was text on the page to indicate the suspension.
        - 'Notification Suspended' was displayed at the top of the page.
        - A date and time were displayed below the login form to tell the user when to expect Random Prompt Notifications to resume (Figure 8).
        - The time displayed did not reflect when the next notification would occur. Instead, it showed the beginning of the next randomized notification period.

3.d.vii Putting Application to Sleep

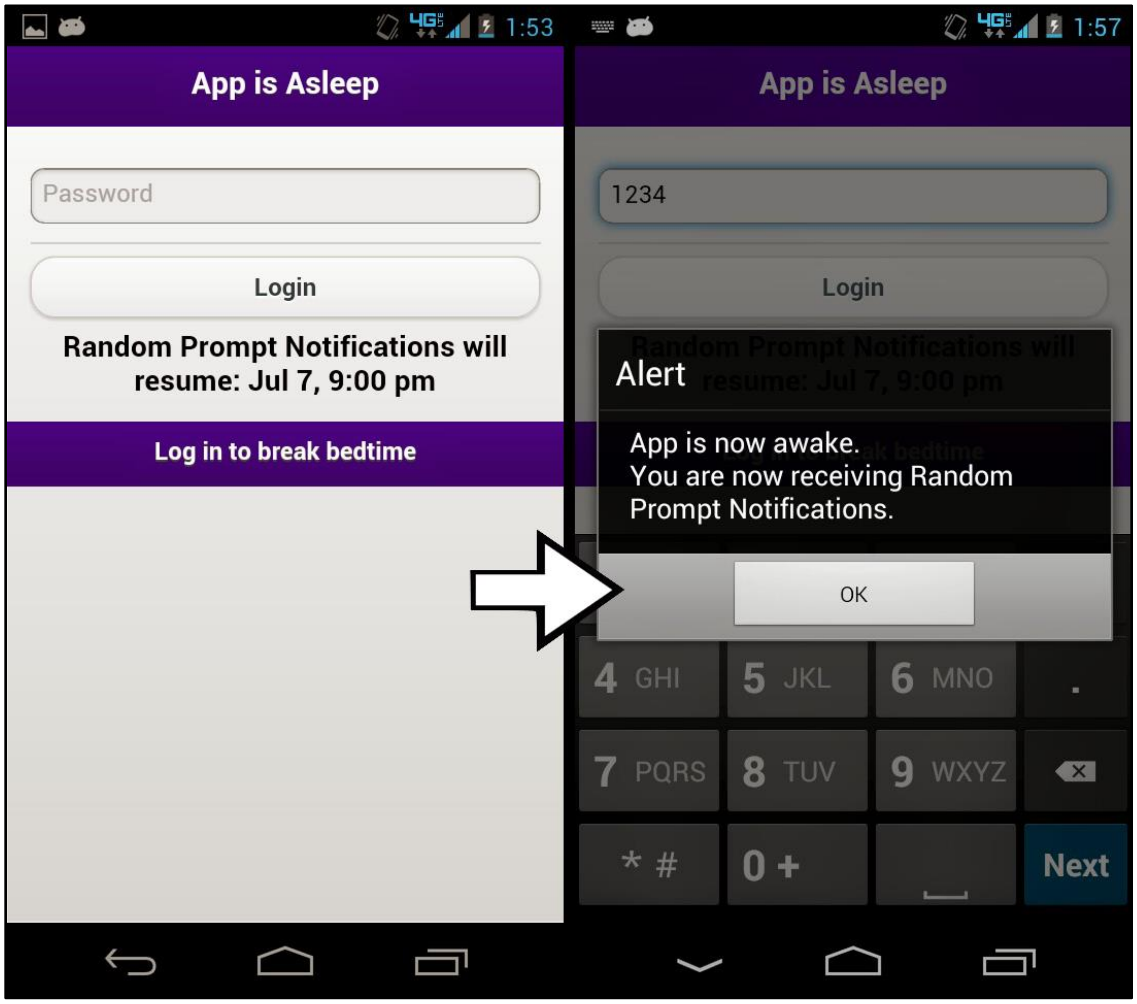


By setting your wake time, you are putting the application to sleep.
Are you sure you want to continue?

Figure 9: Screenshot compositionof the Wake Time alarm configuration page of the e-cigarette survey application; from left to right: initial Wake Time page; setting the wake time with the MobiScroll time scroller widget; waketime confirmation popup.

- A user could put the application to sleep to avoid receiving notifications for an extended period of time. This feature could be accessed from the main menu of the application by tapping 'Set Wake Time'.
- Once a user was within the 'Wake Time' page, he/she could tap on an input field to bring up a three-columned time picker widget (Figure 9).
    - This widget comes from a polished Javascript widget library, *MobiScroll* [14].

    - The time range was in non-military format and the minutes changed in five-minute increments.
    - Once a time was chosen the user had to tap the 'Set' button to continue with the current wake time setting, or 'Cancel' to reset the widget's time and clear the input field.
- Before the application could be put to sleep, the user had to select 'Submit wake time' in the lower right hand corner. This resulted in a pop-up asking the user whether or not they truly wanted to put the application to sleep (Figure 9).
    - If a user chose 'Put to sleep', no further notifications would be sent until the registered wake time was met.
    - If a user chose 'Cancel', then the user would simply be brought back to the 'Wake Time' page.

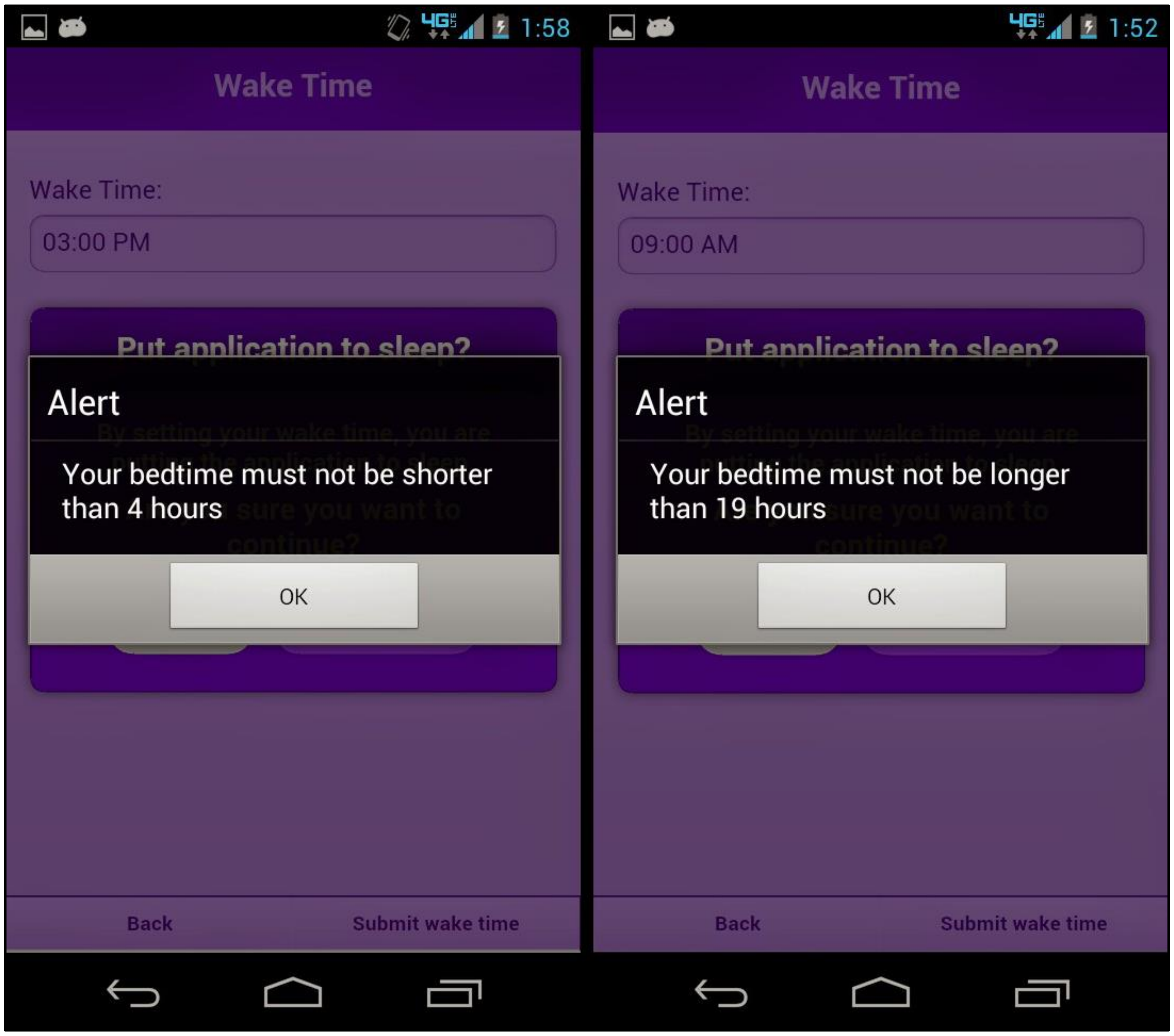


Random Prompt Notifications will resume: Jul 7, 9:00 pm

App is now awake.
You are now receiving Random Prompt Notifications.

Figure 10: Screenshot of the Wake Time function of the e-cigarette survey application. From left to right: resultant login screen after setting a wake time alarm; alert after logging in while application is still asleep.

- Once a user put the application to sleep, the user would be brought to a purple colored login screen (Figure 10).
    - The header, body, and footer text of the login page would change to convey the new status of the login page.
        - ‘App is Asleep’ appeared on the header and ‘Log in to break bedtime’ appeared on the footer of the page.
        - Most importantly, a date and time that the user could expect to receive notifications again appeared below the login button.
- When the Wake Time set by the user had been reached, a unique three-minute long wake time ringtone accompanied the wake time notification, alerting the user that again local notifications could be received.
- The user could interact with the accompanying notification; however, this interaction would not result in the user being able to take a random prompt interview like after other notifications. Instead, the user would be taken to the main menu upon logging in.
- If a user decided to take the app out of sleep at a time earlier than the set Wake Time, the user only had to enter their login information at the login screen.
    - An alert would appear informing the user that the application was now awake and that the user was receiving notifications again (Figure 10).
    - At this time, all previous notifications were cleared and two days of randomized notifications were added.
    - The time span for these new notifications ranged from the time of logging in, until 48 hours later.



Alert

Your bedtime must not be shorter than 4 hours

Alert

Your bedtime must not be longer than 19 hours

Figure 11: From left to right: alert after trying to enter a wake time within 4 fours of the current time; alert after tring to enter a wake time outside of 19 hours from current wake time.

- There were two limitations enforced by the wake time function (Figure 11).
    - The user could not enter a wake time that was less than 4 hours from the time that the user had put the application to sleep.
    - Also, the user could not enter a wake time that was more than 19 hours from the time the user put the app to sleep.
    - Both of these limitations were enforced by alerting the user to the issue.

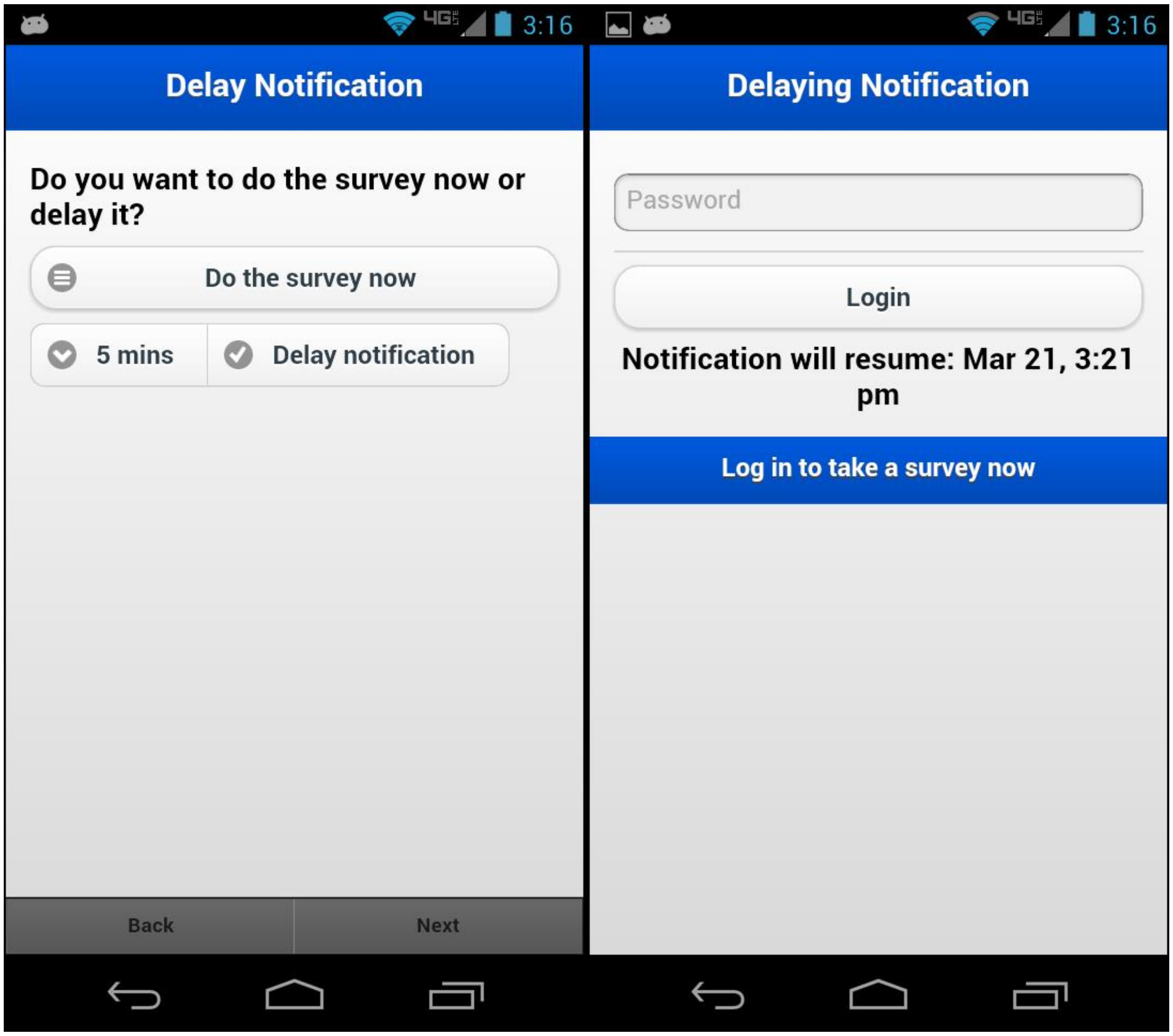

3:16
Delay Notification
Do you want to do the survey now or delay it?
Do the survey now
5 mins
Delay notification
Back
Next
3:16
Delaying Notification
Password
Login
Notification will resume: Mar 21, 3:21 pm
Log in to take a survey now

Figure 12: Screenshot composition of the defunct Delay Notifications capability within the e-cigarette survey application. From left to right: initial Delay Notification prompt; resultant login screen after delaying a notification.

An additional time function was added to the application which allowed a user to delay any notification as it fired (Figure 12). By tapping on a notification, a user would be brought to a screen—prior to the login screen—which would allow them to choose whether or not they wanted to delay the current notification. Not delaying the notification would result in the user having to take a random prompt interview immediately. The user could delay the notification anywhere from 5-30 minutes. Once a user delayed a notification, they would be brought to the login screen where a blue color profile and text would inform the user of the notification being delayed. Due to a necessary rework in notification mechanics within the application, this delay functionality had to be stripped from the final product.

3.d. vii Notifications

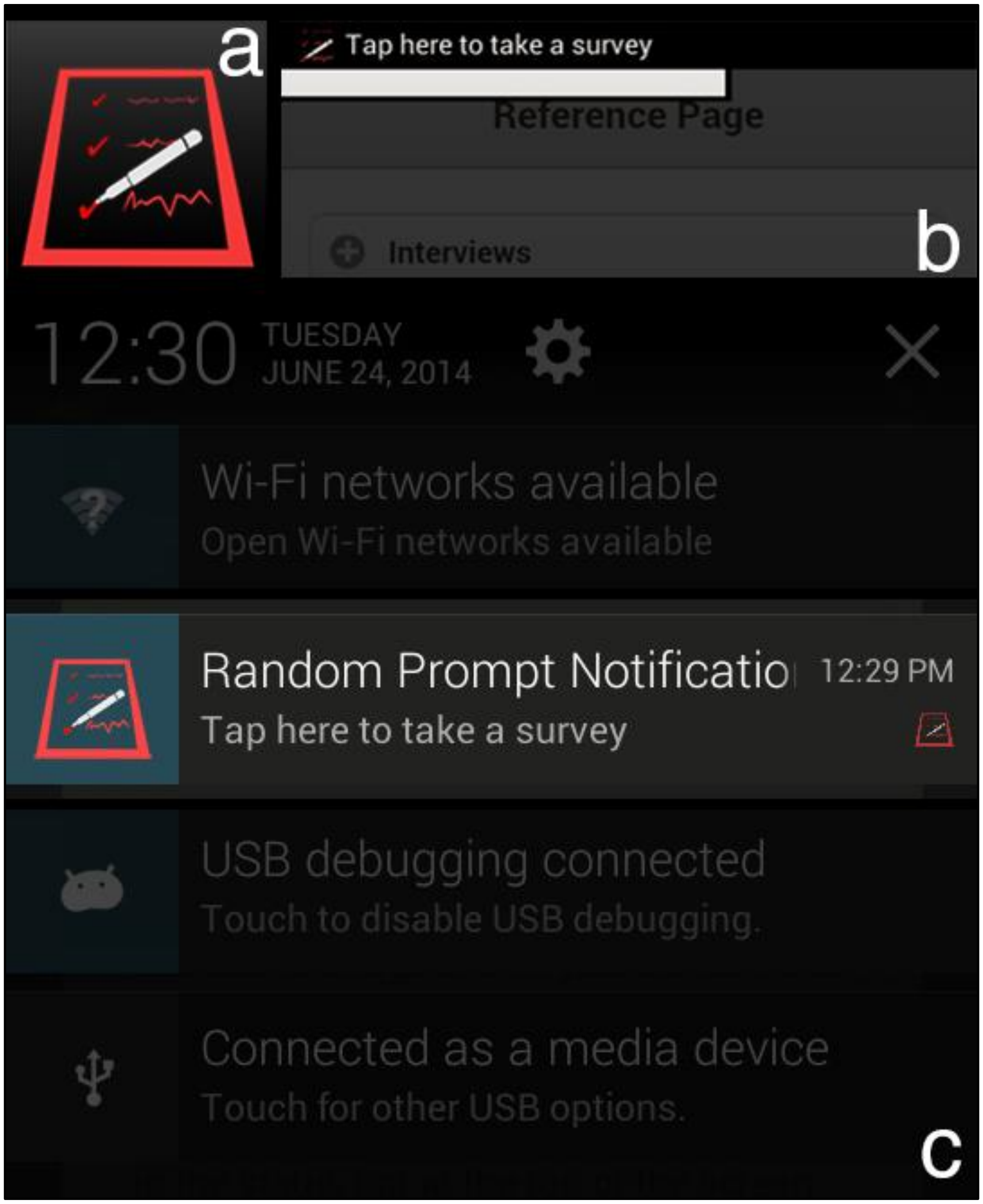

a
Tap here to take a survey
Reference Page
Interviews
b
12:30 TUESDAY JUNE 24, 2014
Wi-Fi networks available
Open Wi-Fi networks available
Random Prompt Notificatio 12:29 PM
Tap here to take a survey
USB debugging connected
Touch to disable USB debugging.
Connected as a media device
Touch for other USB options.
c

Figure 13: Screenshot composition of local notifications within the e-cigarette survey application.(a) Enlarged application & notification icon; (b) Highlighted status bar pop up upon receiving a notification; (c) Highlighted notification message within Android's notificaiton drawer.

3.d.viii.1 Random Prompt Notifications

- A user received up to ten Random Prompt Notifications every 24 hours.
- Each Random Prompt Notification appeared in Android's notification area and featured this icon:
- A three minute long ringtone that increased in pitch every minute accompanied each notification.
- The text contained within the Random Prompt Notification requested that the user tapped on a notification to begin a Random Prompt Interview (Figure 13).
- The time and id of the Random Prompt Notification would also be displayed within the notification.
- Tapping on a Random Prompt Notification took the user to the login page of the application.
- All notifications which populated the Android notification area were cleared only when the user logged into the application.
- If any of the Random Prompt Notifications were older than three minutes, they were recorded as having been missed.
- If a user logged in to the application within three minutes of the latest Random Prompt Notification having fired, they were taken to the beginning of the Random Prompt Interview survey.

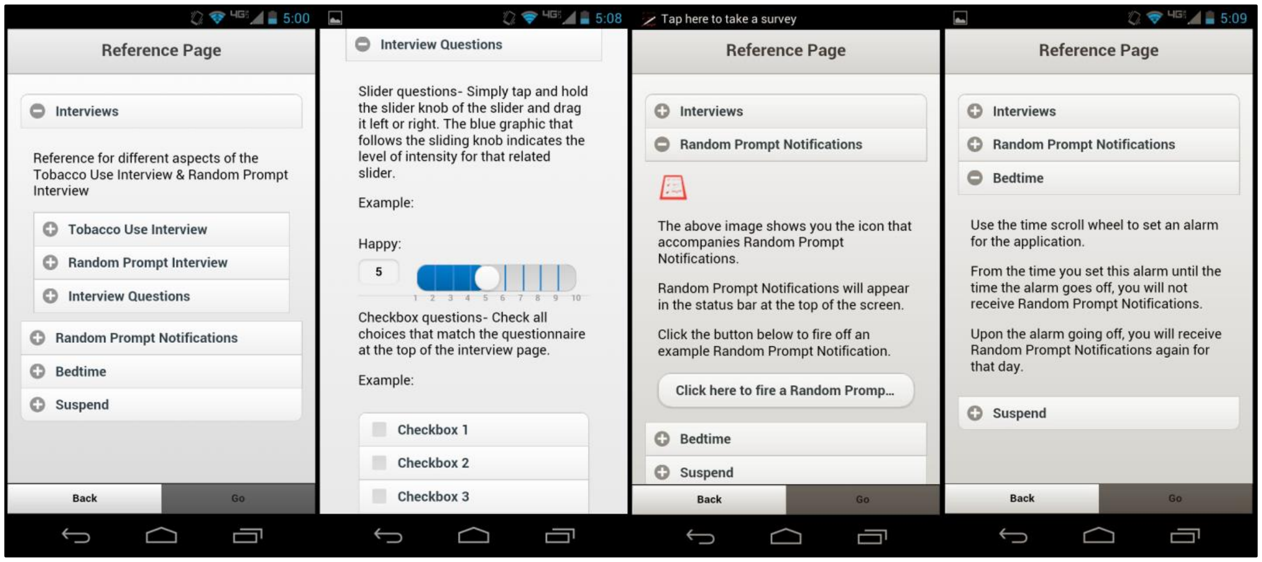
3.d.viii.2 Wake Time Alarm Notifications

- This notification only appeared if a user set a Wake Time Alarm for the application from the main menu.
- Each Wake Time Alarm Notification appeared in Android's notification area and featured this icon:
- A three minute long alarm ringtone accompanied each Wake Time Alarm Notification.
- This ringtone was different than the ringtone that accompanied Random Prompt Notifications with a sound that mirrored a beeping alarm clock.
- The text contained in the Wake Time Alarm Notification prompted the user that the application was now awake.
- The time of the Wake Time Alarm Notification was also displayed within the notification.
- Tapping on a Wake Time Alarm Notification brought the user to the login screen of the application.
- Wake Time Alarm Notifications differed from Random Prompt Notifications in the way they were cancelled.
- Simply tapping on a Wake Time Alarm Notification would cancel the notification.
- There was no connection between Wake Time Alarm Notifications and Random Prompt Notifications.
    - Missing a Wake Time Alarm Notification would not be recorded as the user having missed a notification.
    - Interacting with a Wake Time Alarm Notification would not result in the user having initiated a Random Prompt Interview.

3.d.viii.3 Notification Mechanics

- When a user started the application for the first time, the app populated two days-worth of Random Prompt Notifications for a maximum of 20 notifications.
- If the application was started later in the day, earlier Random Prompt Notifications which would have already occurred were not scheduled.
- This meant that the user usually received less than ten Random Prompt Notifications for that first day.

3.d.ix. Reference Page


5:00

Reference Page

Interviews

Reference for different aspects of the Tobacco Use Interview & Random Prompt Interview

Tobacco Use Interview

Random Prompt Interview

Interview Questions

Random Prompt Notifications

Bedtime

Suspend

Back Go

5:08

Interview Questions

Slider questions- Simply tap and hold the slider knob of the slider and drag it left or right. The blue graphic that follows the sliding knob indicates the level of intensity for that related slider.

Example:

Happy:

5

1 2 3 4 5 6 7 8 9 10

Checkbox questions- Check all choices that match the questionnaire at the top of the interview page.

Example:

Checkbox 1

Checkbox 2

Checkbox 3

Tap here to take a survey

Reference Page

Interviews

Random Prompt Notifications

The above image shows you the icon that accompanies Random Prompt Notifications.

Random Prompt Notifications will appear in the status bar at the top of the screen.

Click the button below to fire off an example Random Prompt Notification.

Click here to fire a Random Promp...

Bedtime

Suspend

Back Go

5:09

Reference Page

Interviews

Random Prompt Notifications

Bedtime

Use the time scroll wheel to set an alarm for the application.

From the time you set this alarm until the time the alarm goes off, you will not receive Random Prompt Notifications.

Upon the alarm going off, you will receive Random Prompt Notifications again for that day.

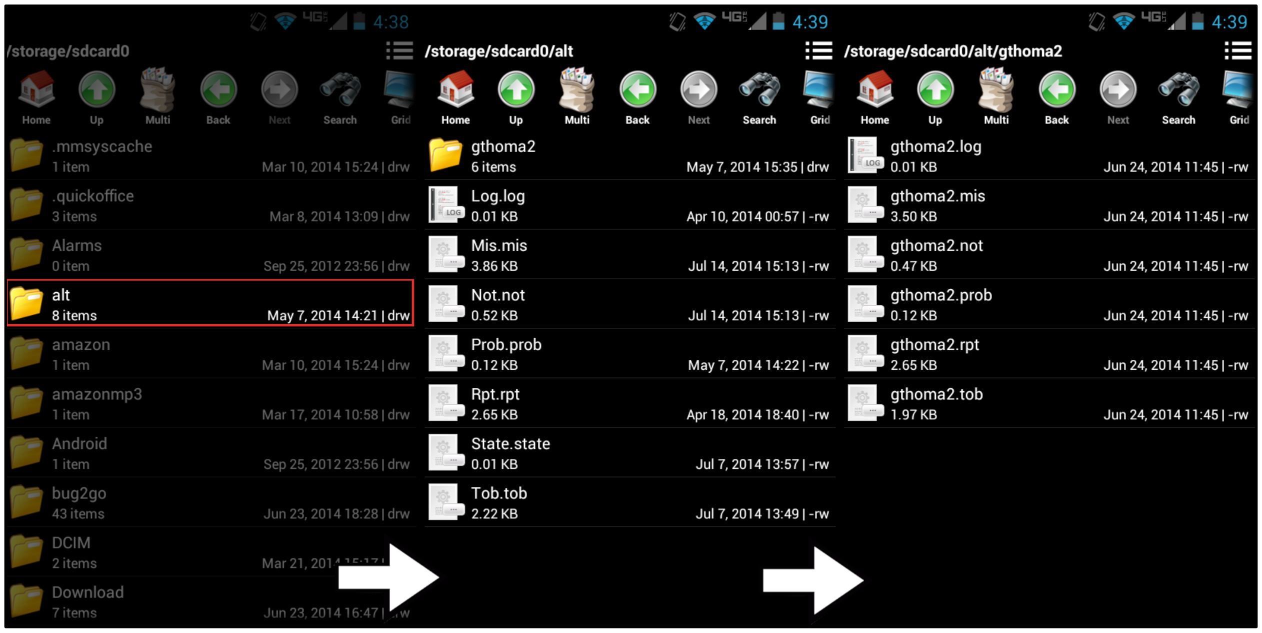
Suspend

Back Go

Figure 14: Screenshot composition of the Reference page within the running e-cigarette survey application; from left to right: dropdown segment showcasing the interviews within the application; dropdown further detailing the question & widget types which can be found within the interviews; dropwdown allowing a user to experience what a notification looks and sounds like; dropdown explaining the bedtime—or Wake Time—function within the application.

To help assist users in using the application, a page was dedicated to displaying different aspects of its performance. The reference section covered the general differences between the two interviews, the intended use of the Wake Time alarm, and the operation of the suspension function. This page also allowed a user to fire off an example notification in order to familiarize the user with future notifications.

3.d.x File Structure



```
/storage/sdcard0
.mmsyscache      1 item     Mar 10, 2014 15:24 | drw
.quickoffice     3 items    Mar 8, 2014 13:09 | drw
Alarms           0 item     Sep 25, 2012 23:56 | drw
alt              8 items    May 7, 2014 14:21 | drw
amazon           1 item     Mar 10, 2014 15:24 | drw
amazonmp3        1 item     Mar 17, 2014 10:58 | drw
Android          1 item     Sep 25, 2012 23:56 | drw
bug2go           43 items   Jun 23, 2014 18:28 | drw
DCIM             2 items    Mar 21, 201
Download         7 items    Jun 23, 2014 16:47 |

/storage/sdcard0/alt
gthoma2          6 items    May 7, 2014 15:35 | drw
Log.log          0.01 KB    Apr 10, 2014 00:57 | -rw
Mis.mis          3.86 KB    Jul 14, 2014 15:13 | -rw
Not.not          0.52 KB    Jul 14, 2014 15:13 | -rw
Prob.prob        0.12 KB    May 7, 2014 14:22 | -rw
Rpt.rpt          2.65 KB    Apr 18, 2014 18:40 | -rw
State.state      0.01 KB    Jul 7, 2014 13:57 | -rw
Tob.tob          2.22 KB    Jul 7, 2014 13:49 | -rw

/storage/sdcard0/alt/gthoma2
gthoma2.log      0.01 KB    Jun 24, 2014 11:45 | -rw
gthoma2.mis      3.50 KB    Jun 24, 2014 11:45 | -rw
gthoma2.not      0.47 KB    Jun 24, 2014 11:45 | -rw
gthoma2.prob     0.12 KB    Jun 24, 2014 11:45 | -rw
gthoma2.rpt      2.65 KB    Jun 24, 2014 11:45 | -rw
gthoma2.tob      1.97 KB    Jun 24, 2014 11:45 | -rw
```

Figure 15: From left to right: From the root directory, a highlighted 'alt' subdirectory containing all user data for the application; inside the 'alt' directory, containing all encrypted data files; within the user directory which contains all relevant decrypted data.

The file structure which contained the data from the application was a simple one. From the root of the Android device, a folder named 'alt' was created on the initial startup of the application. All of the encrypted user data files were stored within 'alt'. All of the names and extensions within this directory reflected the nature of their content. For example: 'log.log' was the file that contained the encrypted contents of the current users login credentials, while 'tob.tob' held the accumulated encrypted contents of that user's tobacco use interviews.

To help distinguish between the active encrypted user data and the prepared decrypted user data, a subdirectory was created upon a researcher tapping 'Prepare Files' within the administrative profile menu (Discussed below). The name of the directory was the current user's user id. All relevant data files from 'alt/…' were copied over to this directory, decrypted, and renamed to the user's id (Figure 15). For example: tob.tob became userID.tob and log.log became userID.log (Appendices D and E).

Due to an odd error in Android's storage system interfacing with Windows Explorer, an extra step was necessary for the research group to grab the processed data off of the phone. If a research member tried to grab the processed data from the newly created, decrypted user directory, the data would not appear from their PC. The solution to this problem was to archive the decrypted user directory. After archiving the user directory, the processed data would be visible within 'alt/…'.

3.d.xi Encryption and Decryption

Because users of the device had free reign to visit the directories where the user data was stored, encrypting & hiding the data from users was essential for the research group. Initially, we discussed nesting the data within sub-directories of the Android system and hiding all files and folders. Unfortunately, this did not perform well as the folders and files were still visible from the Android File Manager app and the method of retrieving the user data proved too complicated for the research group. Instead, the previously mentioned file structure was formed and no folders or files were hidden (Figure 15).

CryptoJS [15] was a Javascript library that contained a plethora of cryptographic algorithms. 128-bit AES encryption & decryption was implemented within the application using *CryptoJS*. Unfortunately, the newline character was used as the active delimiter when decrypting documents with *CryptoJS* and the decryption would only cover the first line of the document. The research group desired decrypted user data with each entry on its own line, so an alternate method of encryption & decryption had to be used.

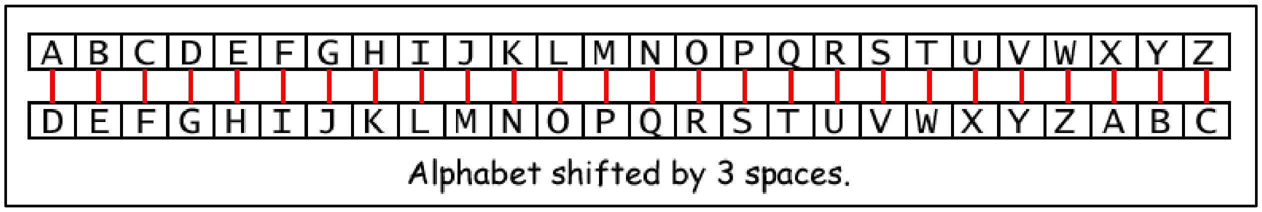

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
D E F G H I J K L M N O P Q R S T U V W X Y Z A B C
Alphabet shifted by 3 spaces.

Figure 16: Diagram exaplaining the Caesar cipher.

Caesar cipher was a simple method of encryption achieved by shifting the place of letters by a certain number of spaces (Figure 16). This was accomplished in the application by increasing the ASCII value of each character by 13. Decryption occurred by decreasing the ASCII value of those characters by the same number. The original Caesar cipher Javascript library used in the application [16] did not incorporate the shift for numbers or punctuation marks, so the open source library was altered to support these additions (Appendices D and E).

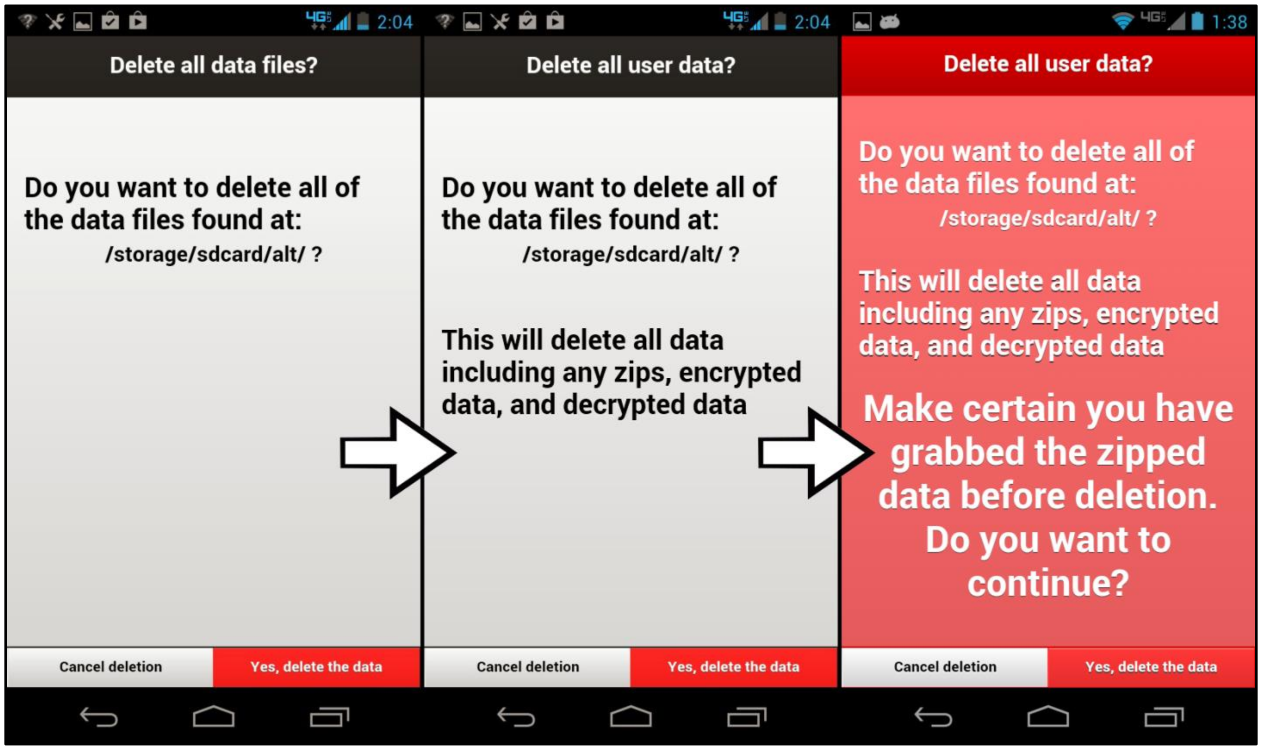
3.d.xii. Administrative Functions

In order for the research group to easily grab all of the user data from each phone, an administrative pin number was hard-coded into the application. Logging into the application with '45459' would navigate the user to an administrative profile page where researchers would be able to 'Prepare Files'. This function was applied to all of the data to be collected by the administrator of the group. An administrator could also delete all of the data currently collected by the user.

3.d.xii.1 Prepare Files

- This function gathered all text files that have had data recorded to them, renamed the files and migrated them according to the file structure outlined in an earlier section of this report.
- As the data within the files were encrypted upon being recorded, the data needed to be decrypted before being read by researchers.
    - A reverse Caesar crypt was applied to the contents of each file.
- Due to a compatibility issue between Android storage devices and the Windows operating system, a user had to archive the created user data folder before being able to grab the data via Windows Explorer.
    - The updated folder of data simply wouldn't appear otherwise.
- The only way to cease all future scheduled notifications from the application was to cancel all of them, so a final function call was made to cancel all future notifications.
    - In the case that this was not the final use of the application, and future notifications were desired, more notifications would be added upon the user logging into the app.

3.d.xii. 2 Delete Files


Delete all data files?

Do you want to delete all of the data files found at:
/storage/sdcard/alt/ ?

Cancel deletion | Yes, delete the data

Delete all user data?

Do you want to delete all of the data files found at:
/storage/sdcard/alt/ ?

This will delete all data including any zips, encrypted data, and decrypted data

Cancel deletion | Yes, delete the data

Delete all user data?

Do you want to delete all of the data files found at:
/storage/sdcard/alt/ ?

This will delete all data including any zips, encrypted data, and decrypted data

Make certain you have grabbed the zipped data before deletion. Do you want to continue?

Cancel deletion | Yes, delete the data

Figure 17: Screenshot composition showing the progression of redundant screens that an administrator must traverse before finally being able to delete all user data.

- This function simply deleted all of the data within the root folder of the application.
- This effectively reset the app so that it would be ready for a new user.
- Deleting the data files through this function call was final, so a warning screen was developed in triplicate. This required a user to click through before actually deleting any data (Figure 17).
- Similar to the 'Prepare Files' function, calling this method also deleted all future scheduled notifications.

3.e Changes and Concessions

At the beginning of the project, the research team outlined their desires for the final product. A responsive and intuitive cross-platform (Android and iOS) application was wanted. Because of this, the group looked into *PhoneGap* as a development platform. *PhoneGap* boasts the ability to more easily cross over mobile applications for Android, iOS, Windows Phone, Blackberry, etc. during development [10].

Unfortunately, the *PhoneGap* package proved to be problematic throughout development. Certain basic needs for the application were difficult to obtain and required outside *plug-ins* in order for them to be realized. The *plug-ins* themselves proved problematic as well, as they were often below version 1.0. Furthermore, each *plug-in* was dependent on the version of Android and *PhoneGap* used during development which introduced a multitude of compatibility issues to the project.

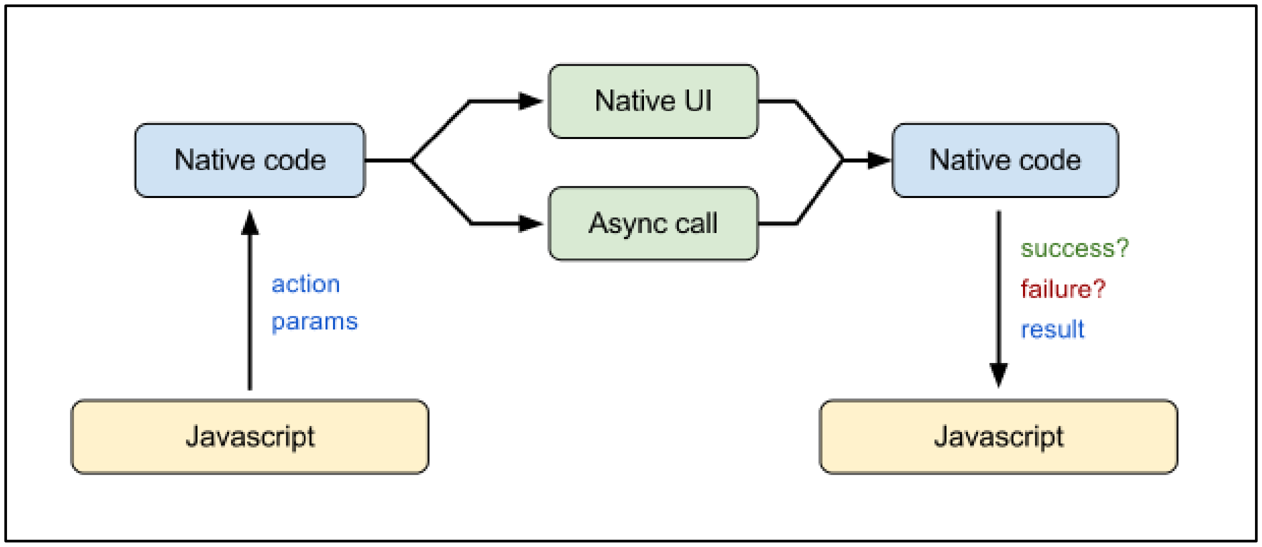


Figure 18: Flow chart of *PhoneGap's* Javascript interface with mobile OS native code.

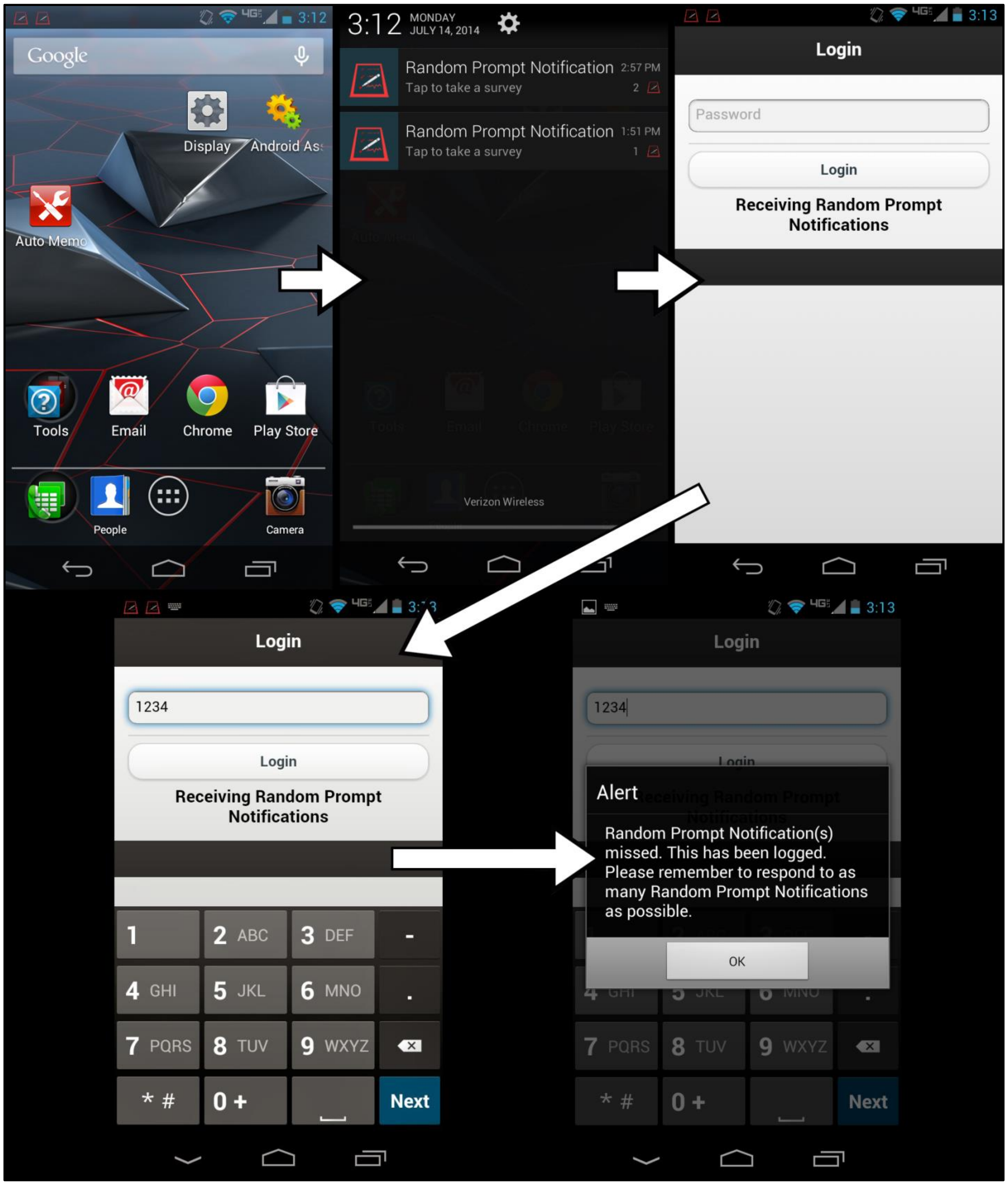
One reoccurring issue during development was coping with the interface between *PhoneGap, PhoneGap* plug-ins, and the asynchronous nature of Javascript code execution (Figure 18). Linear code operation was paramount to many aspects of the application running reliably. The local notification plug-in proved problematic in this regard as many functions within the application required clearing of prior notifications followed by the addition of new notifications. Asynchronous execution sometimes resulted in the addition of notifications in tandem with clearing notifications, which resulted in some or all of the new notifications having been cancelled. To address this particular issue a timer was instated before the addition of notifications so that no notifications would be added until one second had passed since the call to clear prior notifications.

Due to the rigid permission and priority system of the Android framework, some of the features available in the earlier Palm Pilot application were not viable for this project. Short of rooting each phone, the research group could not have absolute control over the volume of the phone nor the ability to stop a user from turning it off. This was remedied with instructions to the user group; they should not diminish the volume of the phone, nor have it turned off.

Each running process within an Android device was given a priority level. The priority level of a process changed over time depending on the amount of memory it was using, how often the process was visited by the user, as well as the nature of the process itself. WebView processes had the lowest priority level of any active process. This meant that when our application was pushed to the background of the Android device, it would inevitably relinquish its space in memory. Because of this, the local notification capability was particularly difficult to get in working order.

The initial method of handling local notifications relied on event listener functions, written in Javascript, to cause desired events upon a notification being added, cancelled, or triggered. The call to cancel a notification after three minutes would be added to the onAdd() event listener, for example. Similarly, the onCancel() event listener housed code that recorded whether or not the notification should be labeled as ‘missed’ by comparing the cancel time of the notification to the trigger time of the notification. Due to the above mentioned Android priority issue, these event listener functions would behave unreliably as the code within would not execute once the application memory was released. The result was that notifications would not faithfully cancel after three minutes, nor would notifications have been recorded as missed.

A major change was made to the application which made local notifications more reliable, more flexible, and still allowed the research group to keep much of the application’s desired performance. Due to the need for this change, however, one facet of the application was obviated; both the UI elements and mechanics behind delayed notifications were omitted. The reason for this omission was again due to the low priority level of the WebView application within the Android system once the application was pushed to the background. The delayed notification mechanic relied on the same event listener functions of the local notifications. Because of this, the delayed notification capability suffered from the same erratic behavior as the initial local notification facility. For example, in order to distinguish a missed notification from one that was delayed, an addendum was made to the body of the onCancel() event function which made a comparison between the time the notification was delayed and the time at which the delay should end, plus three minutes. The result was notifications wrongfully being regarded as not having been missed once the delay period had come and gone.



Random Prompt Notification(s) missed. This has been logged. Please remember to respond to as many Random Prompt Notifications as possible.

Figure 19: Screenshot composition outlining the flow of local notification interaction with regard to the revised notification mechanic. From Left to right & top to bottom: User has received two notificaitons which they have missed; user pulls down notification drawer to tap on a notification; the application is accessed with the login screen appearing first; user enters login pin; user is alerted to having missed notifications and the times at which they were missed is recorded, the missed notifications are cleared. At this time, the user is taken to the main menu of the application.

Due to the above mentioned change, the method for maintaining notifications within Android's notification drawer was altered as well. Luckily these changes resulted in a more intuitive method for the user to handle any missed notifications. Before, notifications and their accompanying ringtone would only last for three minutes, after which they were cancelled and recorded as having been missed if a user did not interact with the notification. The new method of handling notifications resulted in the notifications remaining static within the drawer—they would not disappear after three minutes. This produced a reminder to the user that they had missed a notification. In the case that a user missed multiple notifications, they would be able to observe such, as the notifications would stack within Android's notification drawer, as well as the status bar at the top of the phone. In order to record notifications as having been missed—with regard to this new management of notifications—all of the times that notifications were scheduled to fire were saved to an array of Moment objects—provided by a high-level Javascript Datetime library, *MomentJS* [17]. If—at the time of login— one or more of those times was greater than three minutes ago, the application would record all of those times to file as having been missed and the user would be alerted.

Only one error persisted to the final release of the application; multiple *jQuery Mobile* sliders grouped on the same page would result in a rarely occurring control issue. The issue being that sometimes when a user changed control of one slider handle to another slider, the newly controlled slider would regulate the position and value of the old slider as well. A user could easily fix the issue by controlling the old slider to reselect their old value. Researching and seeking help to fix this problem proved unfruitful as the underlying issue lay with *jQuery* [12], and implementing a UI library which did not rely on *jQuery*—such as *Sencha Touch* 2 [18]— would have resulted in a complete UI overhaul. A fix for the final release of the app increased the performance and responsiveness of the sliders, however. This resulted in a far less frequent occurrence of the problem.

4. Conclusions

By the end of development, the research group had a fully-functional mobile application that allowed them to collect all of the desired data. In addition to reliable data collection, the application boasted convenient functionality for both the users of the study group and members of the research team. The administrative functions allowed members of the research group to retrieve the desired data in a convenient format and file structure. These administrative functions also allowed the research team to quickly and easily turn over the phones used in the study group to a new user. The application provided convenience to the user study group in the form of suspension and wake-time alarm functions which allowed them to cater the app to their schedules when necessary.

The shortcomings of the application were few and born of necessity due to time constraints or the limitations of *PhoneGap*—and *PhoneGap plug-ins*—as a development platform. Unfortunately, taking advantage of *PhoneGap*'s cross-platform capability proved more difficult than advertised and only an Android version of the application was developed. Only one technical error saw its way to release in the application. The frequency of this error was reduced with an interim solution, with an immediate fix to the occurrence of the error being easily realized by the user. Finally, the ancillary function of delaying current notifications was dropped due to the limitations of the local notification plug-in.

5. References

[1] Center for Disease Control, "Notes from the field: electronic cigarette use among middle and high school students-United States," *MMWR. Morbidity and mortality weekly report,* vol. 62, no. 35, p. 729, 2011-2012.

[2] O. D. Flouris AD, "Electronic cigarettes: miracle or menace?," *BMJ,* p. 340:c311, 2010.

[3] J. L. Pearson, et al."E-Cigarette Awareness, Use, and Harm Perceptions in US Adults," *American Hournal of Public Health,* pp. 1758-1766, 2012.

[4] G. S. Dutra LM, "Electronic Cigarettes and Conventional Cigarette Use Among US Adolescents: A Cross-sectional Study," *JAMA Pediactric,* no. 168(7), pp. 610-6617, 2014.

[5] Z. G. Henningfield JE, "Electronic nicotine delivery systems: emerging science foundation for policy," *Tob Control ,* vol. II, no. 19, pp. pp.89-90.

[6] D.A. Axelson, et al."Measuring Mood and Complex Behavior in Natural Environments: Use of Ecological Momentary Assessment in Pediatric Affective Disorders," *Journal of Child and Adolescent Psychopharmacology,* vol. 13, no. 3, pp. 253-266, 2003.

[7] F. B. Turner, et al."Individual and Contextual Influences on Adolescent Smoking," *Annals of the New York Academy of Sciences,* vol. 1021, pp. 175-197, 2004.

[8] R. Mermelstein, et al."Real-time data capture and adolescent cigarette smoking: moods and smoking.," *The Science of Real-Time Data Capture: Self-Report in Health Research,* pp. 117-135, 2007.

[9] Arthur A. Stone, et al."Ecological momentary assessment (EMA) in behavorial medicine," *Annals of Behavioral Medicine,* vol. 16, no. 3, pp. 199-202, 1994.

[10] "Phonegap main page," Adobe, 2014. [Online]. Available: http://phonegap.com/.

[11] "jQuery Mobile 1.3.2 demo showcase," jQuery Mobile, 2013. [Online]. Available: http://demos.jquerymobile.com/1.3.2/.

[12] "jQuery main page," jQuery, 2014. [Online]. Available: http://jquery.com/.

[13] "Motorola Razr M specifications page," 2014. [Online]. Available:

http://www.motorola.com/us/consumers/shop-all-mobile-phones/Droid-Razr-M/m-DROID-RAZR-M.html.

[14] "Mobiscroll main page," Mobiscroll, 2014. [Online]. Available: http://mobiscroll.com.

[15] "CryptoJS v3.1.2 code repository," 2013. [Online]. Available: https://code.google.com/p/crypto-js/.

[16] "Caesar Cipher code repository," 2011. [Online]. Available: http://nayuki.eigenstate.org/page/caesar-cipher-javascript2.

[17] "MomentJS v2.7.0 main page," 2014. [Online]. Available: http://momentjs.com/.

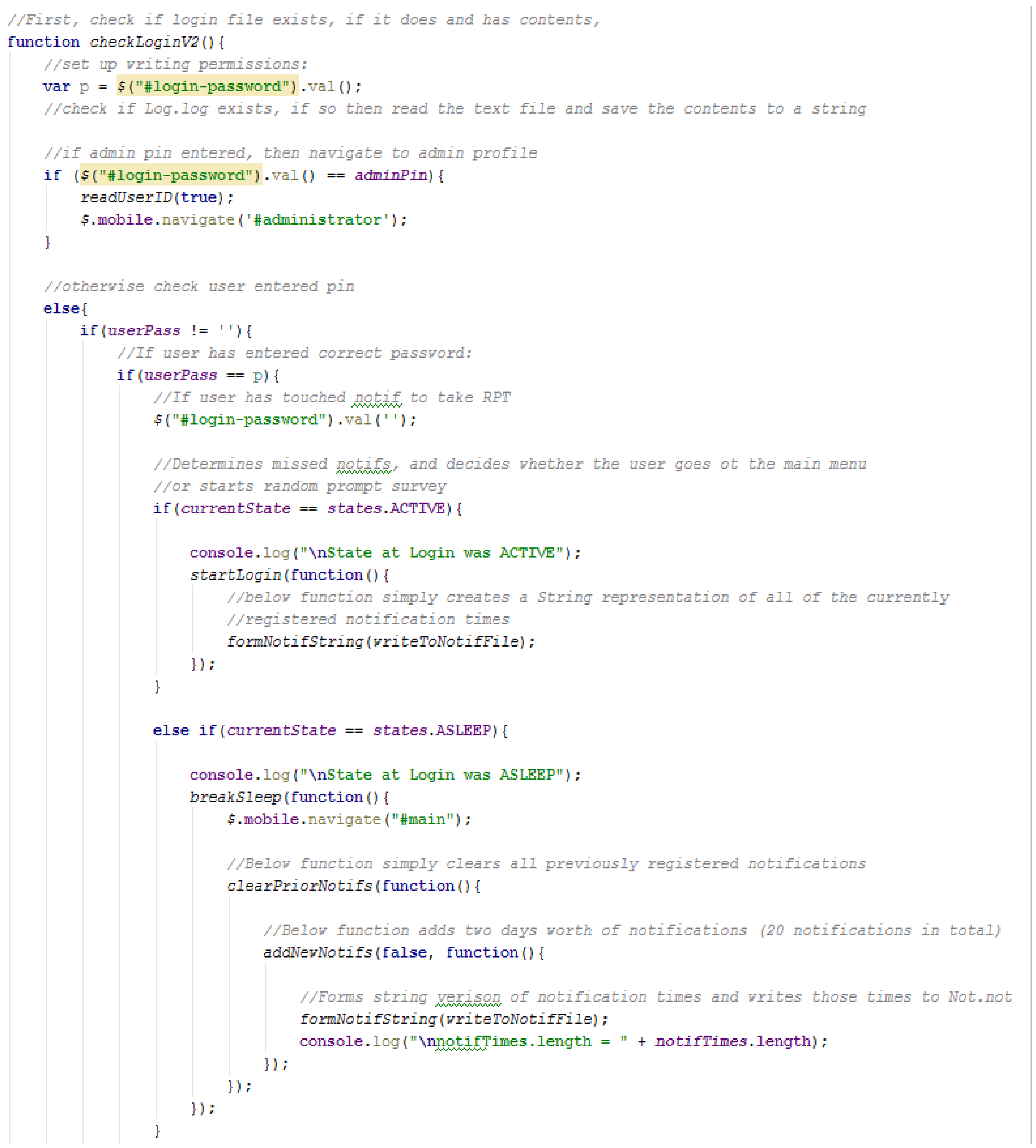
[18] "Sencha Touch 2.0 blog," 2012. [Online]. Available: http://www.sencha.com/blog/announcing-sencha-touch-2/.

6. Appendices

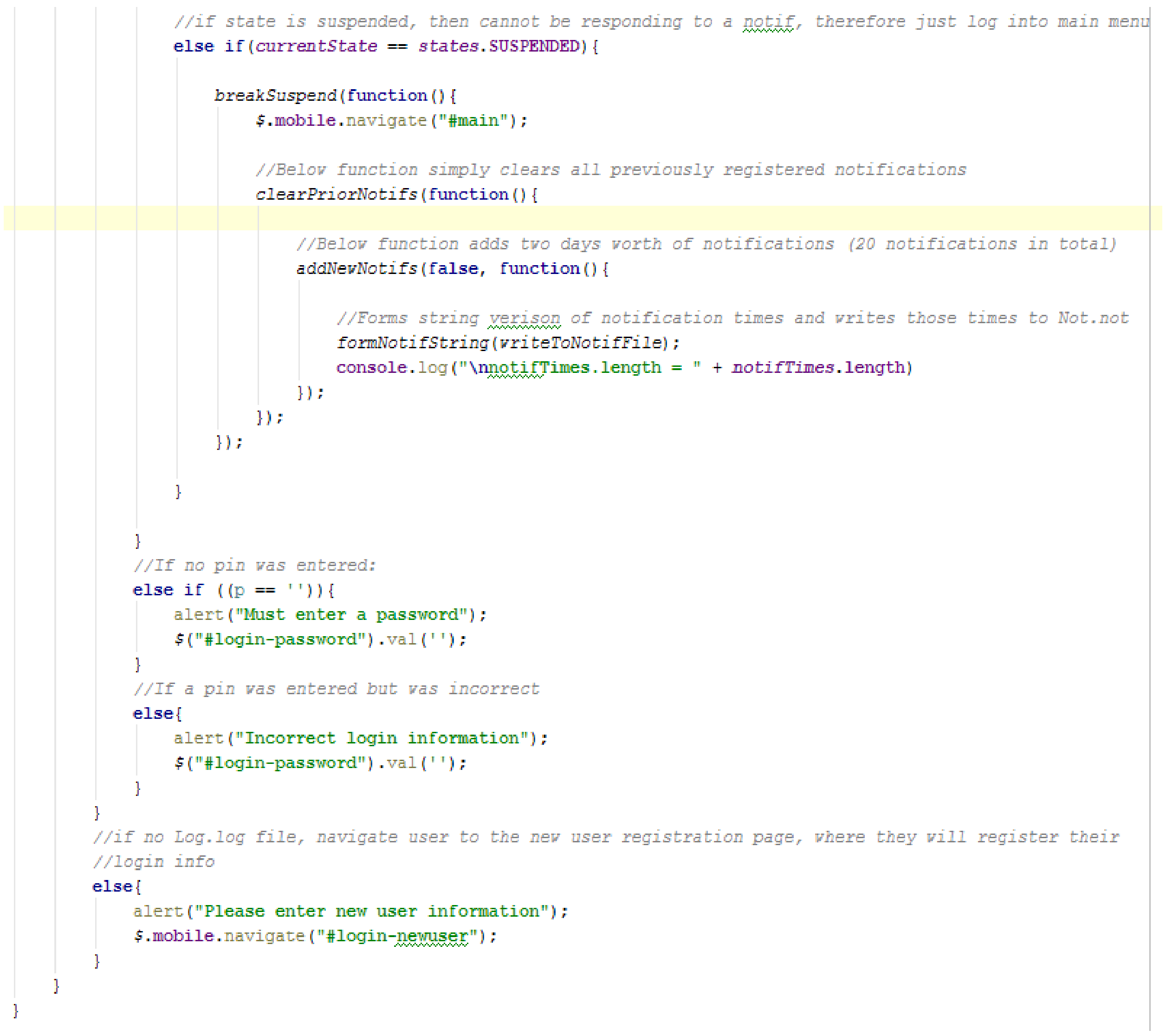
6.A Appendix A Code Snippets:

Appendix A.1 Logging Into the Application:

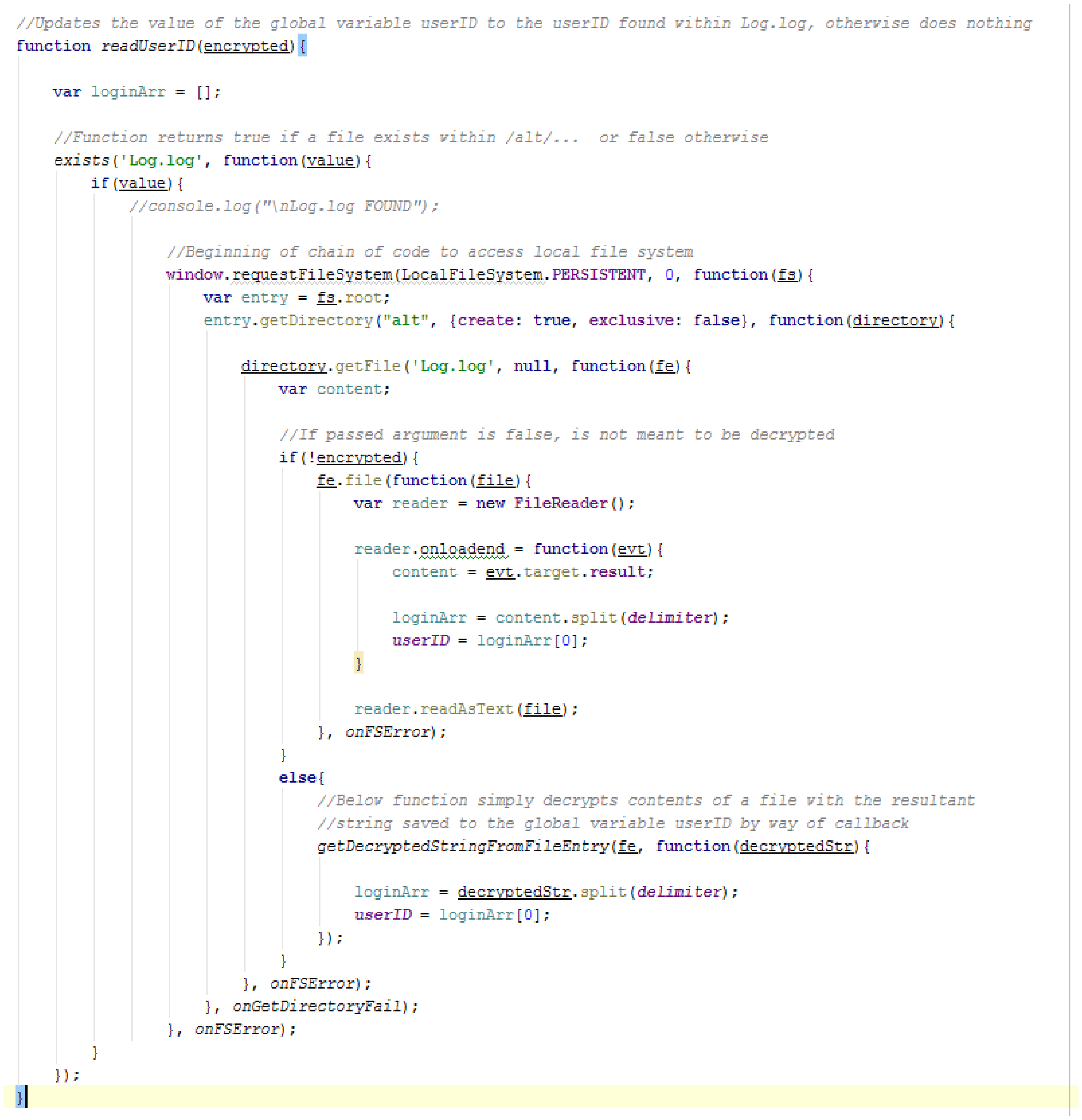
Listed below is the main function, checkLoginV2(), that was used to verify the pin entered from a user while logging in. This function was called when a user tapped the login button. Listed below the main function are all ancillary functions called within checkLoginV2().



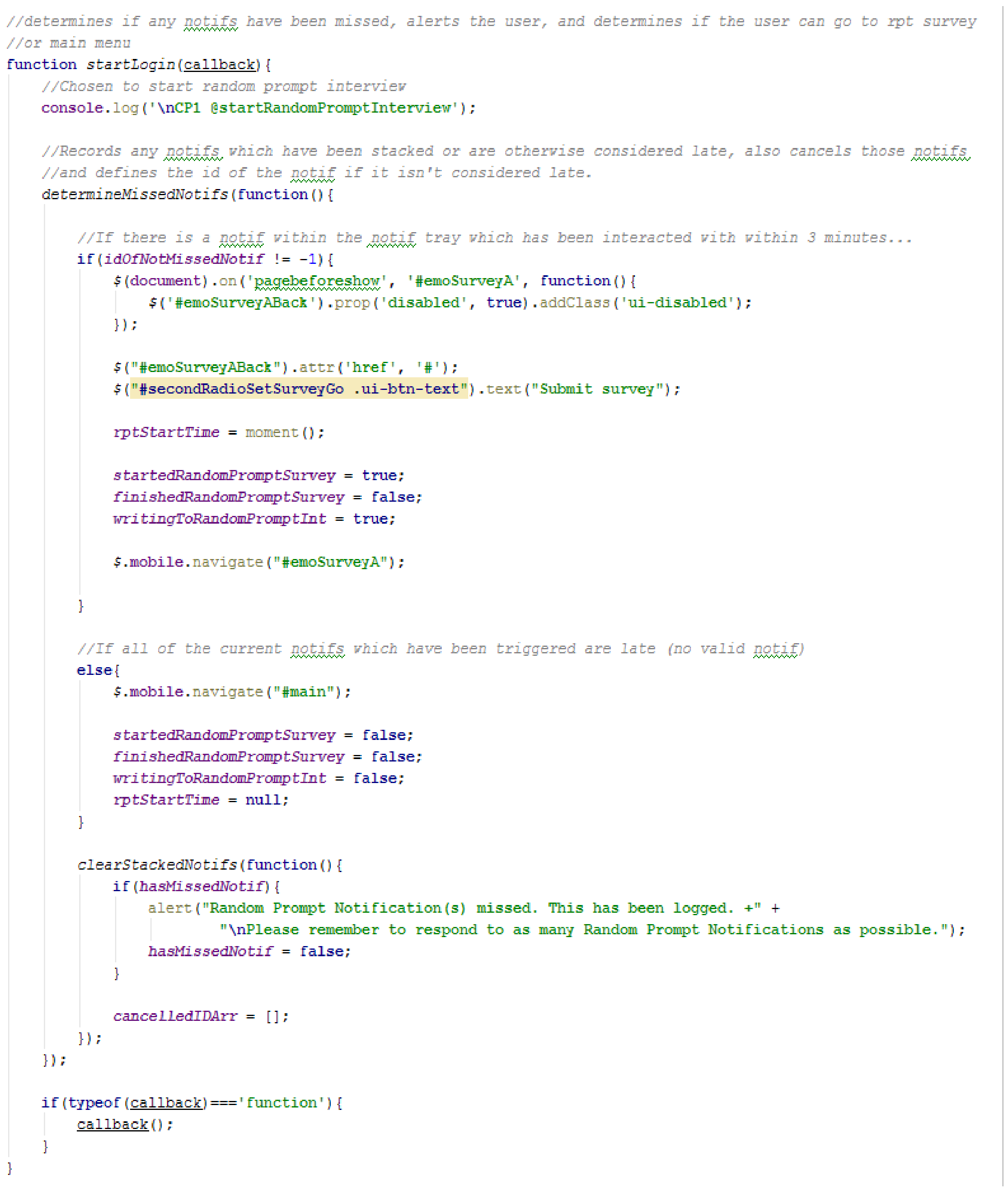
```
//First, check if login file exists, if it does and has contents,
function checkLoginV2(){
    //set up writing permissions:
    var p = $("#login-password").val();
    //check if Log.log exists, if so then read the text file and save the contents to a string

    //if admin pin entered, then navigate to admin profile
    if ($("#login-password").val() == adminPin){
        readUserID(true);
        $.mobile.navigate('#administrator');
    }

    //otherwise check user entered pin
    else{
        if(userPass != ''){
            //If user has entered correct password:
            if(userPass == p){
                //If user has touched notif to take RPT
                $("#login-password").val('');

                //Determines missed notifs, and decides whether the user goes ot the main menu
                //or starts random prompt survey
                if(currentState == states.ACTIVE){

                    console.log("\nState at Login was ACTIVE");
                    startLogin(function(){
                        //below function simply creates a String representation of all of the currently
                        //registered notification times
                        formNotifString(writeToNotifFile);
                    });
                }

                else if(currentState == states.ASLEEP){

                    console.log("\nState at Login was ASLEEP");
                    breakSleep(function(){
                        $.mobile.navigate("#main");

                        //Below function simply clears all previously registered notifications
                        clearPriorNotifs(function(){

                            //Below function adds two days worth of notifications (20 notifications in total)
                            addNewNotifs(false, function(){

                                //Forms string verison of notification times and writes those times to Not.not
                                formNotifString(writeToNotifFile);
                                console.log("\nnotifTimes.length = " + notifTimes.length);
                            });
                        });
                    });
                }
```



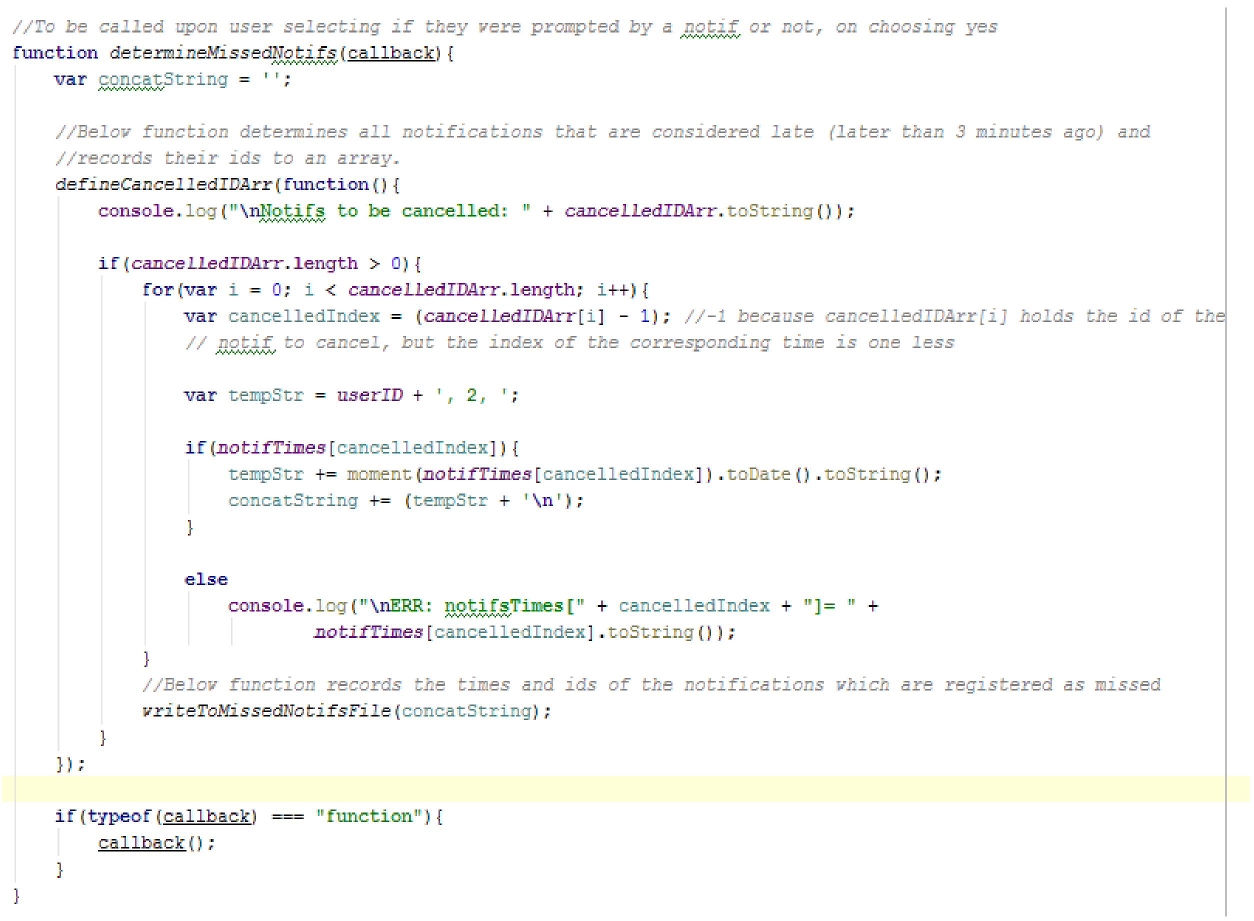
```
                //if state is suspended, then cannot be responding to a notif, therefore just log into main menu
                else if(currentState == states.SUSPENDED){

                    breakSuspend(function(){
                        $.mobile.navigate("#main");

                        //Below function simply clears all previously registered notifications
                        clearPriorNotifs(function(){

                            //Below function adds two days worth of notifications (20 notifications in total)
                            addNewNotifs(false, function(){

                                //Forms string verison of notification times and writes those times to Not.not
                                formNotifString(writeToNotifFile);
                                console.log("\nnotifTimes.length = " + notifTimes.length)
                            });
                        });
                    });

                }

            }
            //If no pin was entered:
            else if ((p == '')){
                alert("Must enter a password");
                $("#login-password").val('');
            }
            //If a pin was entered but was incorrect
            else{
                alert("Incorrect login information");
                $("#login-password").val('');
            }
        }
        //if no Log.log file, navigate user to the new user registration page, where they will register their
        //login info
        else{
            alert("Please enter new user information");
            $.mobile.navigate("#login-newuser");
        }
    }
}
```



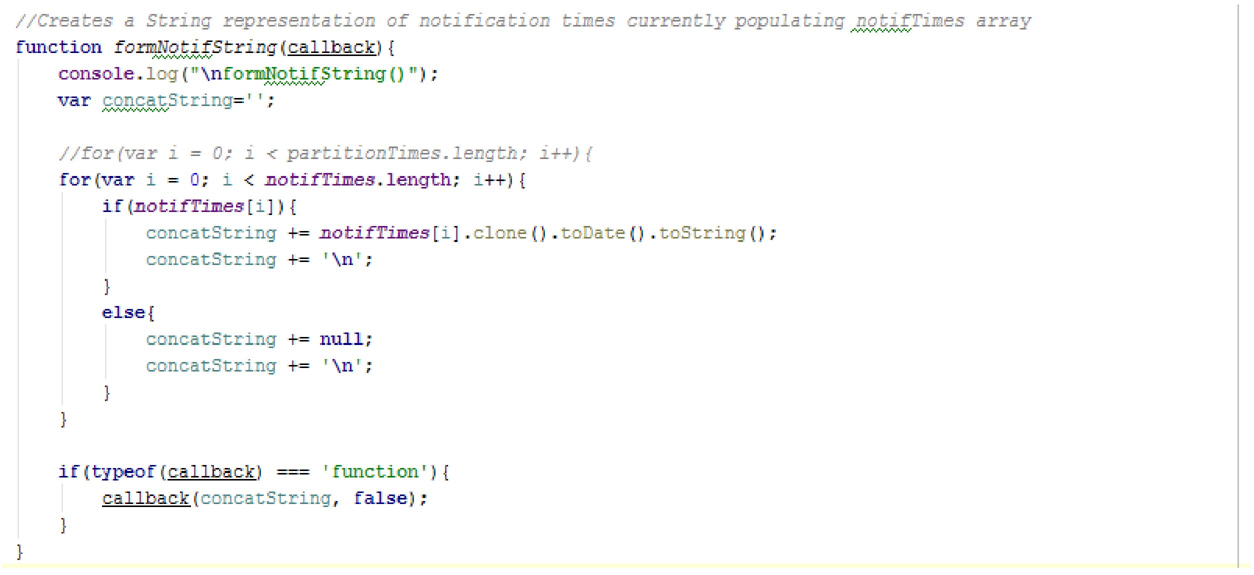
```
//Updates the value of the global variable userID to the userID found within Log.log, otherwise does nothing
function readUserID(encrypted){

    var loginArr = [];

    //Function returns true if a file exists within /alt/...  or false otherwise
    exists('Log.log', function(value){
        if(value){
            //console.log("\nLog.log FOUND");

                //Beginning of chain of code to access local file system
                window.requestFileSystem(LocalFileSystem.PERSISTENT, 0, function(fs){
                    var entry = fs.root;
                    entry.getDirectory("alt", {create: true, exclusive: false}, function(directory){

                        directory.getFile('Log.log', null, function(fe){
                            var content;

                            //If passed argument is false, is not meant to be decrypted
                            if(!encrypted){
                                fe.file(function(file){
                                    var reader = new FileReader();

                                    reader.onloadend = function(evt){
                                        content = evt.target.result;

                                        loginArr = content.split(delimiter);
                                        userID = loginArr[0];
                                    }

                                    reader.readAsText(file);
                                }, onFSError);
                            }
                            else{
                                //Below function simply decrypts contents of a file with the resultant
                                //string saved to the global variable userID by way of callback
                                getDecryptedStringFromFileEntry(fe, function(decryptedStr){

                                    loginArr = decryptedStr.split(delimiter);
                                    userID = loginArr[0];
                                });
                            }
                        }, onFSError);
                    }, onGetDirectoryFail);
                }, onFSError);
        }
    });
}
```



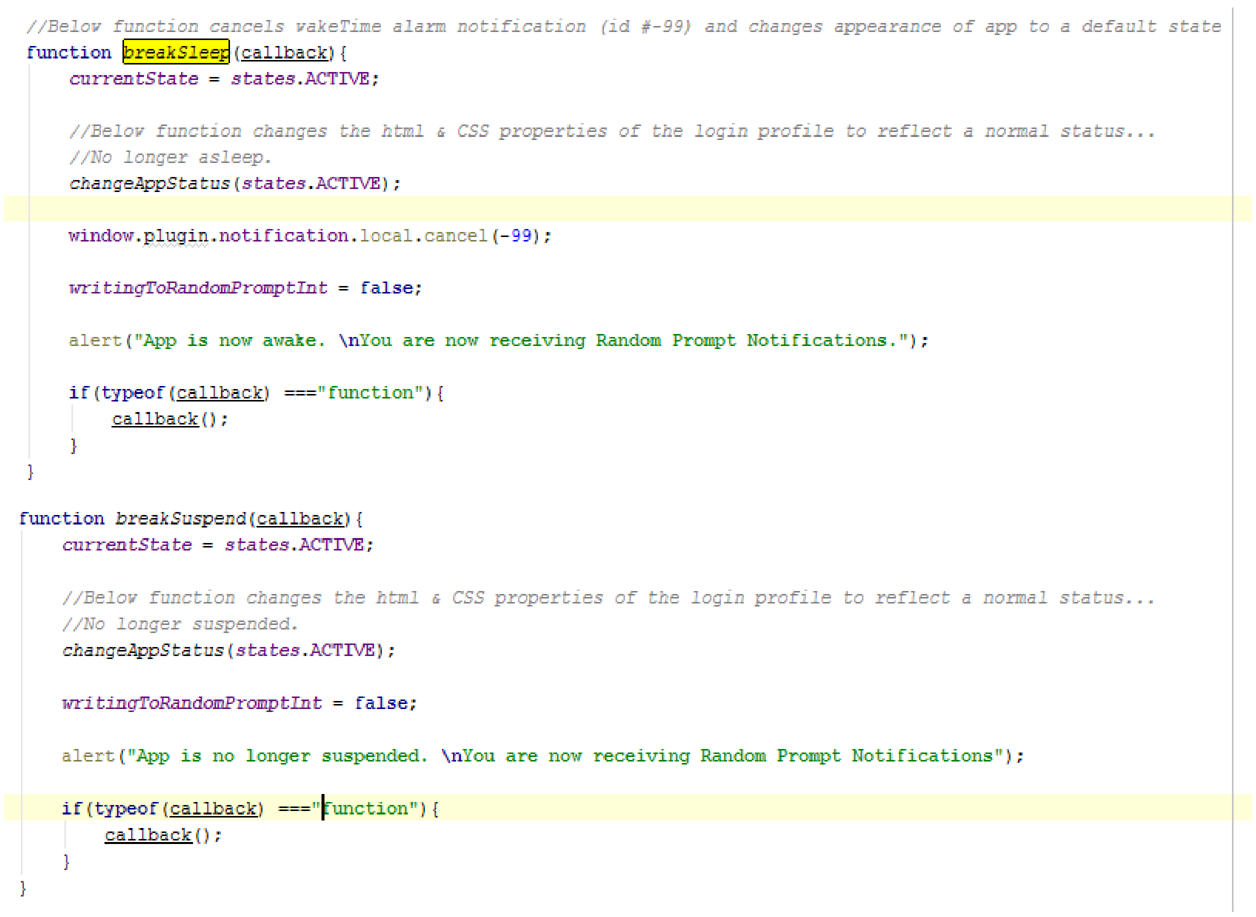
```
//determines if any notifs have been missed, alerts the user, and determines if the user can go to rpt survey
//or main menu
function startLogin(callback){
    //Chosen to start random prompt interview
    console.log('\nCP1 @startRandomPromptInterview');

    //Records any notifs which have been stacked or are otherwise considered late, also cancels those notifs
    //and defines the id of the notif if it isn't considered late.
    determineMissedNotifs(function(){

        //If there is a notif within the notif tray which has been interacted with within 3 minutes...
        if(idOfNotMissedNotif != -1){
            $(document).on('pagebeforeshow', '#emoSurveyA', function(){
                $('#emoSurveyABack').prop('disabled', true).addClass('ui-disabled');
            });

            $("#emoSurveyABack").attr('href', '#');
            $("#secondRadioSetSurveyGo .ui-btn-text").text("Submit survey");

            rptStartTime = moment();

            startedRandomPromptSurvey = true;
            finishedRandomPromptSurvey = false;
            writingToRandomPromptInt = true;

            $.mobile.navigate("#emoSurveyA");

        }

        //If all of the current notifs which have been triggered are late (no valid notif)
        else{
            $.mobile.navigate("#main");

            startedRandomPromptSurvey = false;
            finishedRandomPromptSurvey = false;
            writingToRandomPromptInt = false;
            rptStartTime = null;
        }

        clearStackedNotifs(function(){
            if(hasMissedNotif){
                alert("Random Prompt Notification(s) missed. This has been logged. +" +
                        "\nPlease remember to respond to as many Random Prompt Notifications as possible.");
                hasMissedNotif = false;
            }

            cancelledIDArr = [];
        });
    });

    if(typeof(callback)==='function'){
        callback();
    }
}
```



```
//To be called upon user selecting if they were prompted by a notif or not, on choosing yes
function determineMissedNotifs(callback){
    var concatString = '';

    //Below function determines all notifications that are considered late (later than 3 minutes ago) and
    //records their ids to an array.
    defineCancelledIDArr(function(){
        console.log("\nNotifs to be cancelled: " + cancelledIDArr.toString());

        if(cancelledIDArr.length > 0){
            for(var i = 0; i < cancelledIDArr.length; i++){
                var cancelledIndex = (cancelledIDArr[i] - 1); //-1 because cancelledIDArr[i] holds the id of the
                // notif to cancel, but the index of the corresponding time is one less

                var tempStr = userID + ', 2, ';

                if(notifTimes[cancelledIndex]){
                    tempStr += moment(notifTimes[cancelledIndex]).toDate().toString();
                    concatString += (tempStr + '\n');
                }

                else
                    console.log("\nERR: notifsTimes[" + cancelledIndex + "]= " +
                            notifTimes[cancelledIndex].toString());
            }
            //Below function records the times and ids of the notifications which are registered as missed
            writeToMissedNotifsFile(concatString);
        }
    });

    if(typeof(callback) === "function"){
        callback();
    }
}
```



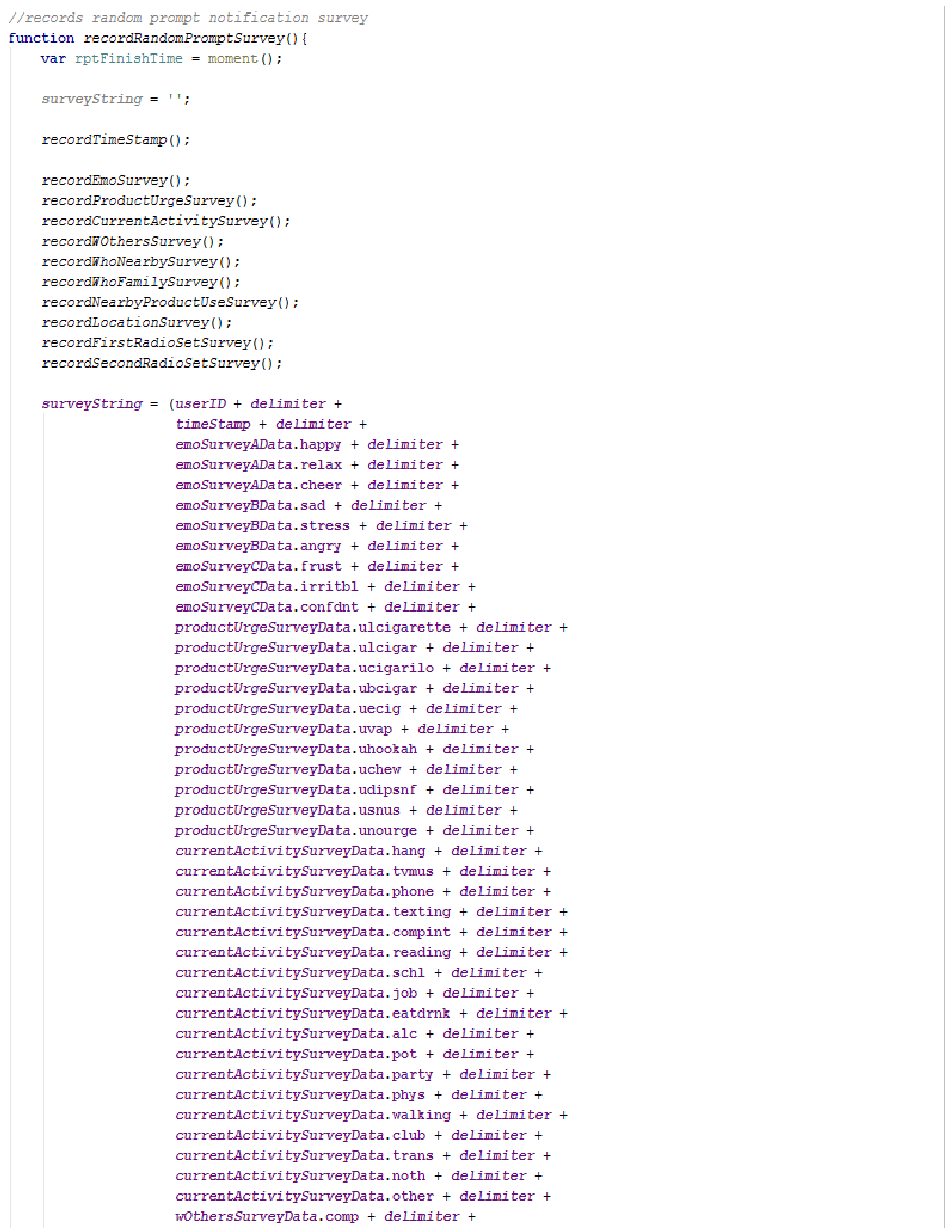
```
//Creates a String representation of notification times currently populating notifTimes array
function formNotifString(callback){
    console.log("\nformNotifString()");
    var concatString='';

    //for(var i = 0; i < partitionTimes.length; i++){
    for(var i = 0; i < notifTimes.length; i++){
        if(notifTimes[i]){
            concatString += notifTimes[i].clone().toDate().toString();
            concatString += '\n';
        }
        else{
            concatString += null;
            concatString += '\n';
        }
    }

    if(typeof(callback) === 'function'){
        callback(concatString, false);
    }
}
```



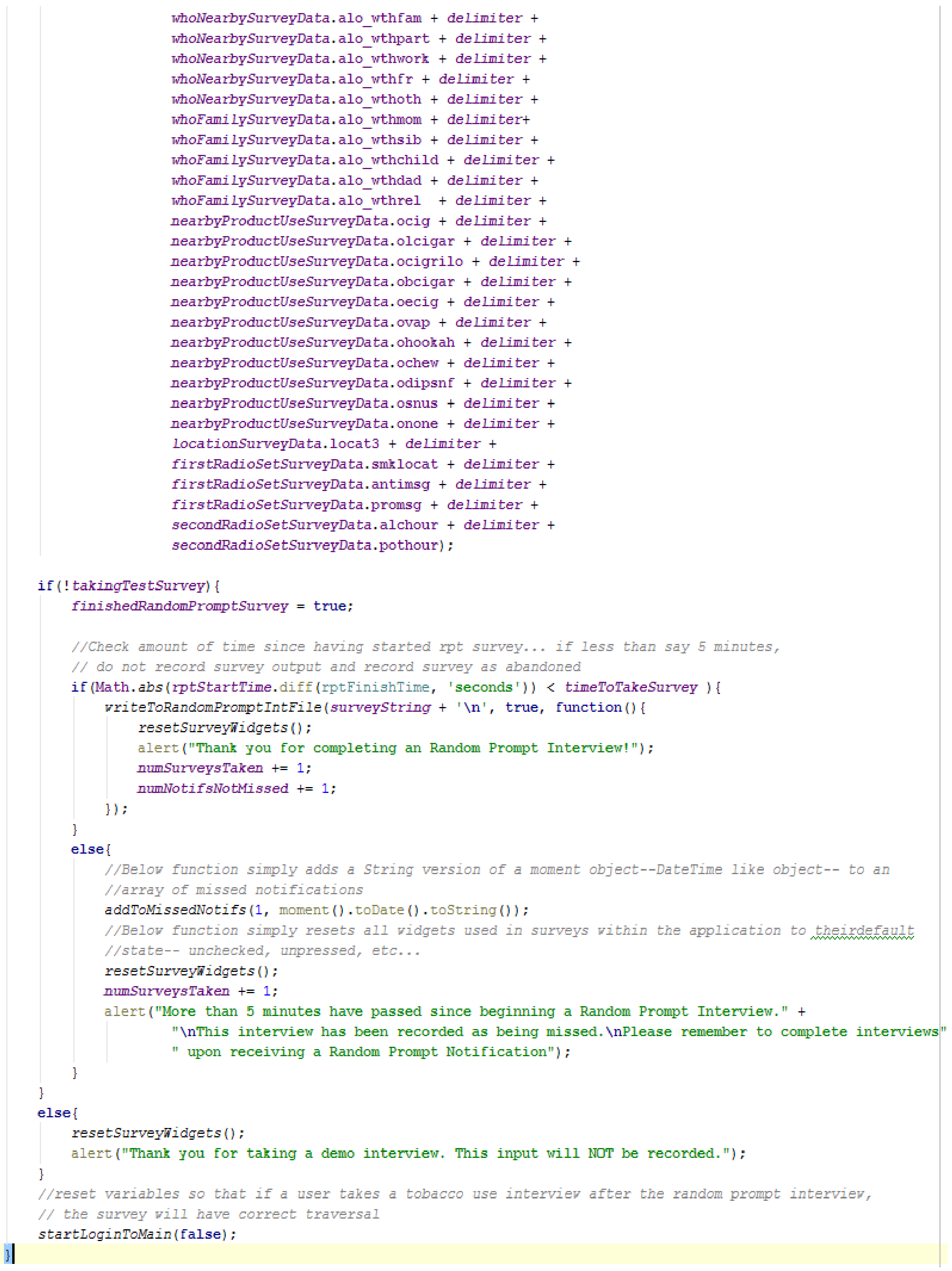
```
//Below function cancels wakeTime alarm notification (id #-99) and changes appearance of app to a default state
function breakSleep(callback){
    currentState = states.ACTIVE;

    //Below function changes the html & CSS properties of the login profile to reflect a normal status...
    //No longer asleep.
    changeAppStatus(states.ACTIVE);

    window.plugin.notification.local.cancel(-99);

    writingToRandomPromptInt = false;

    alert("App is now awake. \nYou are now receiving Random Prompt Notifications.");

    if(typeof(callback) ==="function"){
        callback();
    }
}

function breakSuspend(callback){
    currentState = states.ACTIVE;

    //Below function changes the html & CSS properties of the login profile to reflect a normal status...
    //No longer suspended.
    changeAppStatus(states.ACTIVE);

    writingToRandomPromptInt = false;

    alert("App is no longer suspended. \nYou are now receiving Random Prompt Notifications");

    if(typeof(callback) ==="function"){
        callback();
    }
}
```

Appendix A.2 Submitting Interview Answers:

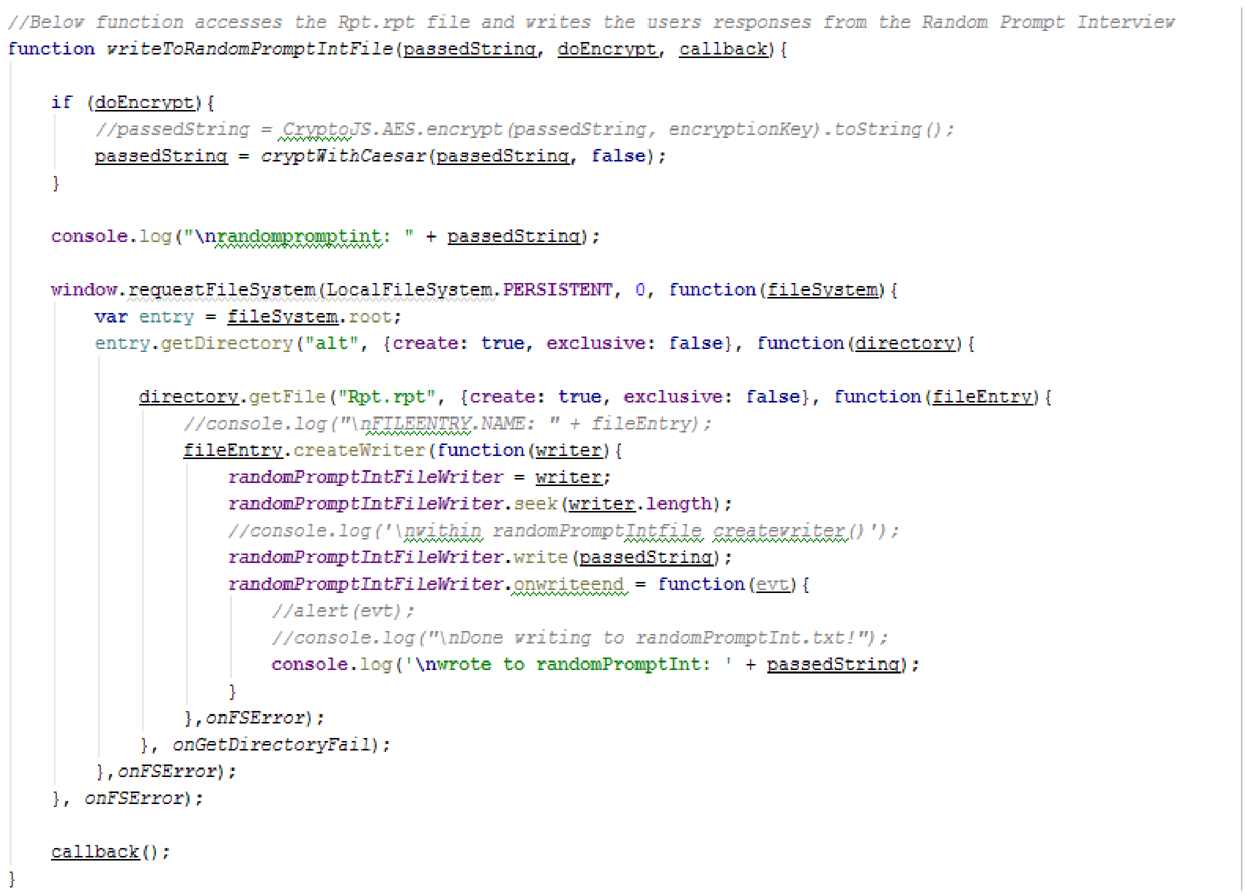
Listed below are the functions used to record interview answers for the Random Prompt Interviews. Functions used to record interview answers for the Tobacco Use Interview were not displayed because they followed the same code structure as the functions used to record Random Prompt Interviews.



```
//records random prompt notification survey
function recordRandomPromptSurvey(){
    var rptFinishTime = moment();

    surveyString = '';

    recordTimeStamp();

    recordEmoSurvey();
    recordProductUrgeSurvey();
    recordCurrentActivitySurvey();
    recordWOthersSurvey();
    recordWhoNearbySurvey();
    recordWhoFamilySurvey();
    recordNearbyProductUseSurvey();
    recordLocationSurvey();
    recordFirstRadioSetSurvey();
    recordSecondRadioSetSurvey();

    surveyString = (userID + delimiter +
                    timeStamp + delimiter +
                    emoSurveyAData.happy + delimiter +
                    emoSurveyAData.relax + delimiter +
                    emoSurveyAData.cheer + delimiter +
                    emoSurveyBData.sad + delimiter +
                    emoSurveyBData.stress + delimiter +
                    emoSurveyBData.angry + delimiter +
                    emoSurveyCData.frust + delimiter +
                    emoSurveyCData.irritbl + delimiter +
                    emoSurveyCData.confdnt + delimiter +
                    productUrgeSurveyData.ulcigarette + delimiter +
                    productUrgeSurveyData.ulcigar + delimiter +
                    productUrgeSurveyData.ucigarilo + delimiter +
                    productUrgeSurveyData.ubcigar + delimiter +
                    productUrgeSurveyData.uecig + delimiter +
                    productUrgeSurveyData.uvap + delimiter +
                    productUrgeSurveyData.uhookah + delimiter +
                    productUrgeSurveyData.uchew + delimiter +
                    productUrgeSurveyData.udipsnf + delimiter +
                    productUrgeSurveyData.usnus + delimiter +
                    productUrgeSurveyData.unourge + delimiter +
                    currentActivitySurveyData.hang + delimiter +
                    currentActivitySurveyData.tvmus + delimiter +
                    currentActivitySurveyData.phone + delimiter +
                    currentActivitySurveyData.texting + delimiter +
                    currentActivitySurveyData.compint + delimiter +
                    currentActivitySurveyData.reading + delimiter +
                    currentActivitySurveyData.schl + delimiter +
                    currentActivitySurveyData.job + delimiter +
                    currentActivitySurveyData.eatdrnk + delimiter +
                    currentActivitySurveyData.alc + delimiter +
                    currentActivitySurveyData.pot + delimiter +
                    currentActivitySurveyData.party + delimiter +
                    currentActivitySurveyData.phys + delimiter +
                    currentActivitySurveyData.walking + delimiter +
                    currentActivitySurveyData.club + delimiter +
                    currentActivitySurveyData.trans + delimiter +
                    currentActivitySurveyData.noth + delimiter +
                    currentActivitySurveyData.other + delimiter +
                    wOthersSurveyData.comp + delimiter +
```



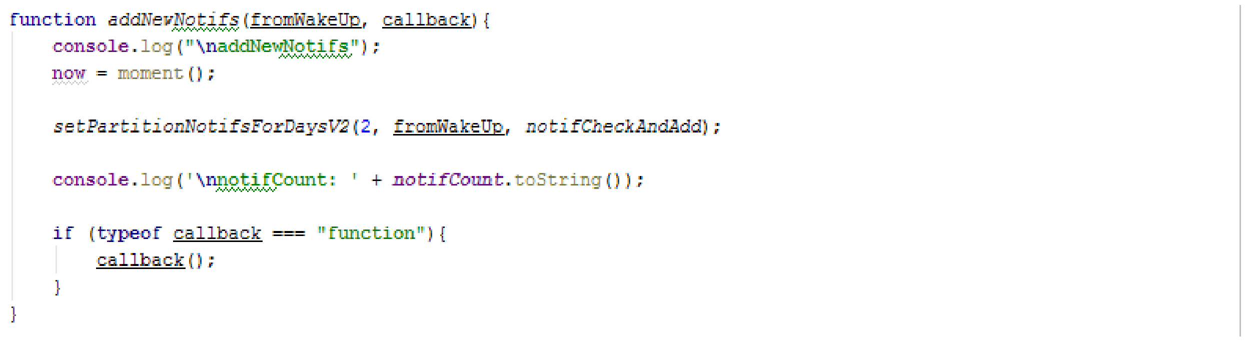
```
                whoNearbySurveyData.alo_wthfam + delimiter +
                whoNearbySurveyData.alo_wthpart + delimiter +
                whoNearbySurveyData.alo_wthwork + delimiter +
                whoNearbySurveyData.alo_wthfr + delimiter +
                whoNearbySurveyData.alo_wthoth + delimiter +
                whoFamilySurveyData.alo_wthmom + delimiter+
                whoFamilySurveyData.alo_wthsib + delimiter +
                whoFamilySurveyData.alo_wthchild + delimiter +
                whoFamilySurveyData.alo_wthdad + delimiter +
                whoFamilySurveyData.alo_wthrel  + delimiter +
                nearbyProductUseSurveyData.ocig + delimiter +
                nearbyProductUseSurveyData.olcigar + delimiter +
                nearbyProductUseSurveyData.ocigrilo + delimiter +
                nearbyProductUseSurveyData.obcigar + delimiter +
                nearbyProductUseSurveyData.oecig + delimiter +
                nearbyProductUseSurveyData.ovap + delimiter +
                nearbyProductUseSurveyData.ohookah + delimiter +
                nearbyProductUseSurveyData.ochew + delimiter +
                nearbyProductUseSurveyData.odipsnf + delimiter +
                nearbyProductUseSurveyData.osnus + delimiter +
                nearbyProductUseSurveyData.onone + delimiter +
                locationSurveyData.locat3 + delimiter +
                firstRadioSetSurveyData.smklocat + delimiter +
                firstRadioSetSurveyData.antimsg + delimiter +
                firstRadioSetSurveyData.promsg + delimiter +
                secondRadioSetSurveyData.alchour + delimiter +
                secondRadioSetSurveyData.pothour);

    if(!takingTestSurvey){
        finishedRandomPromptSurvey = true;

        //Check amount of time since having started rpt survey... if less than say 5 minutes,
        // do not record survey output and record survey as abandoned
        if(Math.abs(rptStartTime.diff(rptFinishTime, 'seconds')) < timeToTakeSurvey ){
            writeToRandomPromptIntFile(surveyString + '\n', true, function(){
                resetSurveyWidgets();
                alert("Thank you for completing an Random Prompt Interview!");
                numSurveysTaken += 1;
                numNotifsNotMissed += 1;
            });
        }
        else{
            //Below function simply adds a String version of a moment object--DateTime like object-- to an
            //array of missed notifications
            addToMissedNotifs(1, moment().toDate().toString());
            //Below function simply resets all widgets used in surveys within the application to theirdefault
            //state-- unchecked, unpressed, etc...
            resetSurveyWidgets();
            numSurveysTaken += 1;
            alert("More than 5 minutes have passed since beginning a Random Prompt Interview." +
                    "\nThis interview has been recorded as being missed.\nPlease remember to complete interviews"
                    " upon receiving a Random Prompt Notification");
        }
    }
    else{
        resetSurveyWidgets();
        alert("Thank you for taking a demo interview. This input will NOT be recorded.");
    }
    //reset variables so that if a user takes a tobacco use interview after the random prompt interview,
    // the survey will have correct traversal
    startLoginToMain(false);
}
```



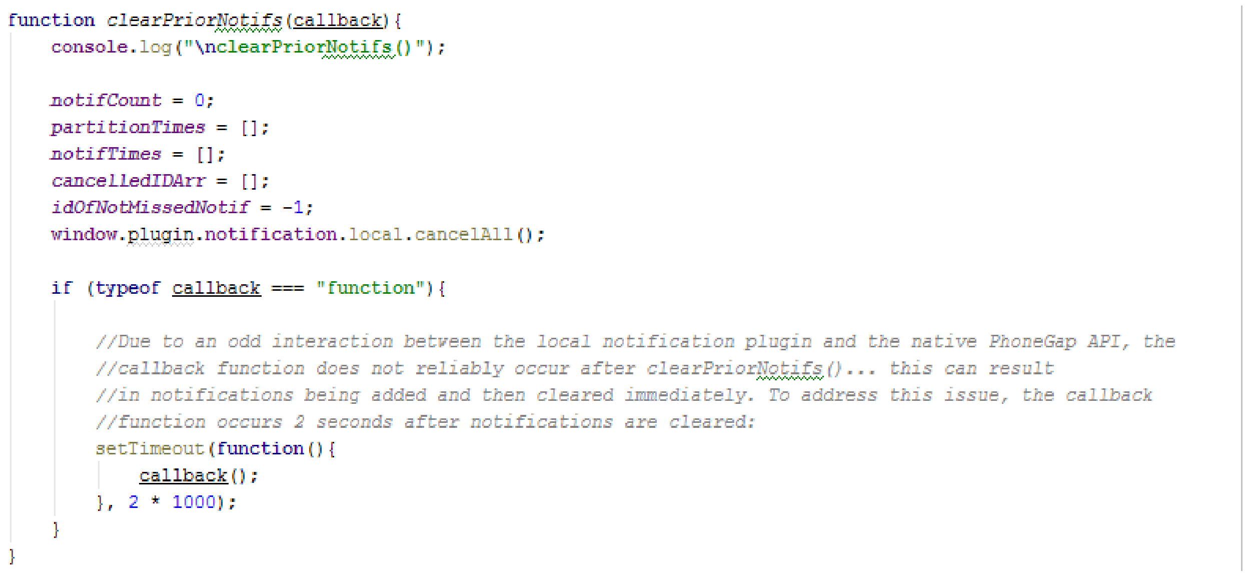
```
//Below function accesses the Rpt.rpt file and writes the users responses from the Random Prompt Interview
function writeToRandomPromptIntFile(passedString, doEncrypt, callback){

    if (doEncrypt){
        //passedString = CryptoJS.AES.encrypt(passedString, encryptionKey).toString();
        passedString = cryptWithCaesar(passedString, false);
    }

    console.log("\nrandompromptint: " + passedString);

    window.requestFileSystem(LocalFileSystem.PERSISTENT, 0, function(fileSystem){
        var entry = fileSystem.root;
        entry.getDirectory("alt", {create: true, exclusive: false}, function(directory){

            directory.getFile("Rpt.rpt", {create: true, exclusive: false}, function(fileEntry){
                //console.log("\nFILEENTRY.NAME: " + fileEntry);
                fileEntry.createWriter(function(writer){
                    randomPromptIntFileWriter = writer;
                    randomPromptIntFileWriter.seek(writer.length);
                    //console.log('\nwithin randomPromptIntfile createwriter()');
                    randomPromptIntFileWriter.write(passedString);
                    randomPromptIntFileWriter.onwriteend = function(evt){
                        //alert(evt);
                        //console.log("\nDone writing to randomPromptInt.txt!");
                        console.log('\nwrote to randomPromptInt: ' + passedString);
                    }
                },onFSError);
            }, onGetDirectoryFail);
        },onFSError);
    }, onFSError);

    callback();
}
```

Appendix A.3 Resetting and Adding Daily Notifications:

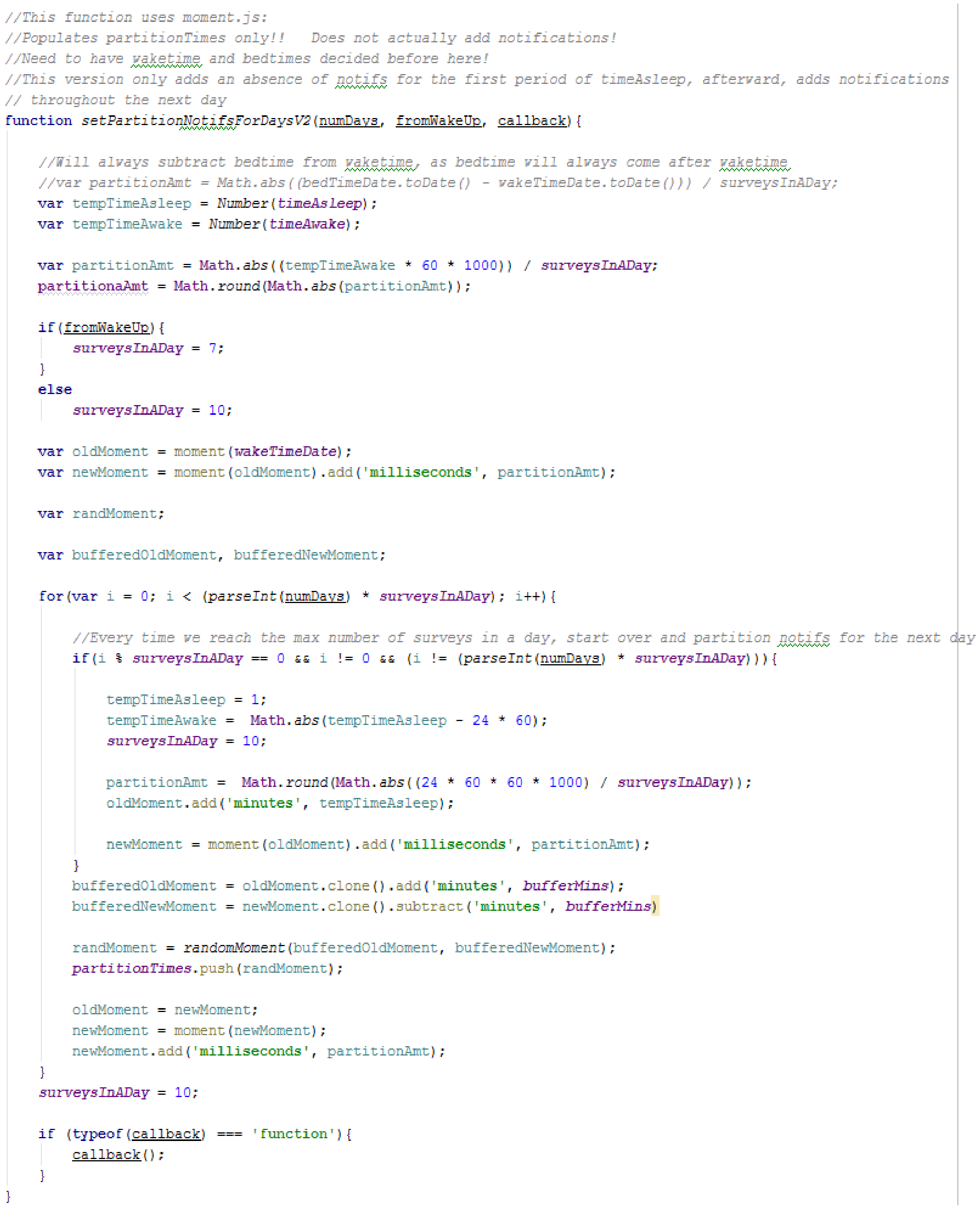
The following functions were used to clear prior notifications and add new notifications. This combination of code was used throughout the application and was the only way to update notifications. addNewNotifs() was always passed as a callback argument to clearPriorNotifs(), with the result being that no notifications would be added until the old notifications were first cleared.



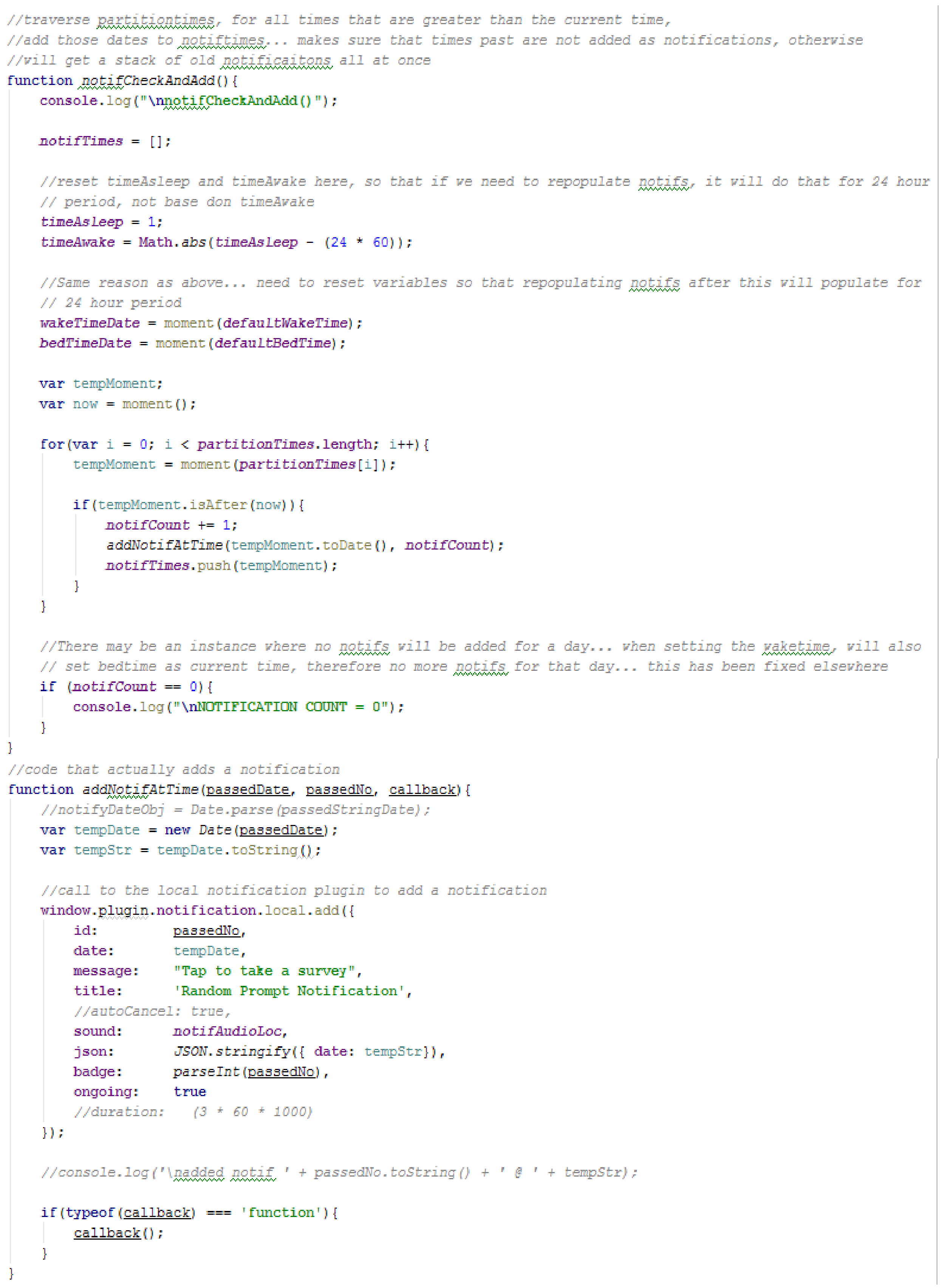
```
function addNewNotifs(fromWakeUp, callback){
    console.log("\naddNewNotifs");
    now = moment();

    setPartitionNotifsForDaysV2(2, fromWakeUp, notifCheckAndAdd);

    console.log('\nnotifCount: ' + notifCount.toString());

    if (typeof callback === "function"){
        callback();
    }
}
```



```
function clearPriorNotifs(callback){
    console.log("\nclearPriorNotifs()");

    notifCount = 0;
    partitionTimes = [];
    notifTimes = [];
    cancelledIDArr = [];
    idOfNotMissedNotif = -1;
    window.plugin.notification.local.cancelAll();

    if (typeof callback === "function"){

        //Due to an odd interaction between the local notification plugin and the native PhoneGap API, the
        //callback function does not reliably occur after clearPriorNotifs()... this can result
        //in notifications being added and then cleared immediately. To address this issue, the callback
        //function occurs 2 seconds after notifications are cleared:
        setTimeout(function(){
            callback();
        }, 2 * 1000);
    }
}
```



```
//This function uses moment.js:
//Populates partitionTimes only!!   Does not actually add notifications!
//Need to have waketime and bedtimes decided before here!
//This version only adds an absence of notifs for the first period of timeAsleep, afterward, adds notifications
// throughout the next day
function setPartitionNotifsForDaysV2(numDays, fromWakeUp, callback){

    //Will always subtract bedtime from waketime, as bedtime will always come after waketime
    //var partitionAmt = Math.abs((bedTimeDate.toDate() - wakeTimeDate.toDate())) / surveysInADay;
    var tempTimeAsleep = Number(timeAsleep);
    var tempTimeAwake = Number(timeAwake);

    var partitionAmt = Math.abs((tempTimeAwake * 60 * 1000)) / surveysInADay;
    partitionaAmt = Math.round(Math.abs(partitionAmt));

    if(fromWakeUp){
        surveysInADay = 7;
    }
    else
        surveysInADay = 10;

    var oldMoment = moment(wakeTimeDate);
    var newMoment = moment(oldMoment).add('milliseconds', partitionAmt);

    var randMoment;

    var bufferedOldMoment, bufferedNewMoment;

    for(var i = 0; i < (parseInt(numDays) * surveysInADay); i++){

        //Every time we reach the max number of surveys in a day, start over and partition notifs for the next day
        if(i % surveysInADay == 0 && i != 0 && (i != (parseInt(numDays) * surveysInADay))){

            tempTimeAsleep = 1;
            tempTimeAwake =  Math.abs(tempTimeAsleep - 24 * 60);
            surveysInADay = 10;

            partitionAmt =  Math.round(Math.abs((24 * 60 * 60 * 1000) / surveysInADay));
            oldMoment.add('minutes', tempTimeAsleep);

            newMoment = moment(oldMoment).add('milliseconds', partitionAmt);
        }
        bufferedOldMoment = oldMoment.clone().add('minutes', bufferMins);
        bufferedNewMoment = newMoment.clone().subtract('minutes', bufferMins)

        randMoment = randomMoment(bufferedOldMoment, bufferedNewMoment);
        partitionTimes.push(randMoment);

        oldMoment = newMoment;
        newMoment = moment(newMoment);
        newMoment.add('milliseconds', partitionAmt);
    }
    surveysInADay = 10;

    if (typeof(callback) === 'function'){
        callback();
    }
}
```



```
//traverse partitiontimes, for all times that are greater than the current time,
//add those dates to notiftimes... makes sure that times past are not added as notifications, otherwise
//will get a stack of old notificaitons all at once
function notifCheckAndAdd(){
    console.log("\nnotifCheckAndAdd()");

    notifTimes = [];

    //reset timeAsleep and timeAwake here, so that if we need to repopulate notifs, it will do that for 24 hour
    // period, not base don timeAwake
    timeAsleep = 1;
    timeAwake = Math.abs(timeAsleep - (24 * 60));

    //Same reason as above... need to reset variables so that repopulating notifs after this will populate for
    // 24 hour period
    wakeTimeDate = moment(defaultWakeTime);
    bedTimeDate = moment(defaultBedTime);

    var tempMoment;
    var now = moment();

    for(var i = 0; i < partitionTimes.length; i++){
        tempMoment = moment(partitionTimes[i]);

        if(tempMoment.isAfter(now)){
            notifCount += 1;
            addNotifAtTime(tempMoment.toDate(), notifCount);
            notifTimes.push(tempMoment);
        }
    }

    //There may be an instance where no notifs will be added for a day... when setting the waketime, will also
    // set bedtime as current time, therefore no more notifs for that day... this has been fixed elsewhere
    if (notifCount == 0){
        console.log("\nNOTIFICATION COUNT = 0");
    }
}
//code that actually adds a notification
function addNotifAtTime(passedDate, passedNo, callback){
    //notifyDateObj = Date.parse(passedStringDate);
    var tempDate = new Date(passedDate);
    var tempStr = tempDate.toString();

    //call to the local notification plugin to add a notification
    window.plugin.notification.local.add({
        id:         passedNo,
        date:       tempDate,
        message:    "Tap to take a survey",
        title:      'Random Prompt Notification',
        //autoCancel: true,
        sound:      notifAudioLoc,
        json:       JSON.stringify({ date: tempStr}),
        badge:      parseInt(passedNo),
        ongoing:    true
        //duration:   (3 * 60 * 1000)
    });

    //console.log('\nadded notif ' + passedNo.toString() + ' @ ' + tempStr);

    if(typeof(callback) === 'function'){
        callback();
    }
}
```

6.B Appendix B Questions for the Tobacco Use Interview

The set of questions listed below were extracted directly from the FDA Communications Ecological Momentary Assessment Interviews Codebook (Appendix F.1). This document was authored by members of the research group: Dr. Robin Mermelstein and John O'Keefe. This particular document is an altered form of a previous document which had been used by the research group for over 10 years in similar tobacco use studies—including the *PalmPilot* study mentioned above. Outlined below are the sets of questions featured in the **Tobacco Use Interview** within the E-cigarette Survey Application.

1. What type of tobacco product did you use? TUProduct

Cigarette	___ 1	Vaporizer	___ 6
Little Cigar	___ 2	Hookah	___ 7
Cigarillo	___ 3	Chew	___ 8
Big Cigar	___ 4	Dip/Snuff	___ 9
e-cigarette	___ 5	Snus	___ 10

1a2. *If* TUProduct =1 **(Cigarette)**
For cigarette, was the cigarette menthol? CIGMENT

Yes ___ 2 No ___ 1

2. Think about how you feel right now…

			Not at All				Somewhat					Very Much
2a.	Right now: I feel Happy	HAPPY	1	2	3	4	5	6	7	8	9	10
2b.	Right now: I feel Relaxed	RELAX	1	2	3	4	5	6	7	8	9	10
2c.	Right now: I feel Sad	SAD	1	2	3	4	5	6	7	8	9	10
2d.	Right now: I feel Stressed	STRESS	1	2	3	4	5	6	7	8	9	10
2e.	Right now: I feel Angry	ANGRY	1	2	3	4	5	6	7	8	9	10
2f.	Right now: I feel Cheerful	CHEER	1	2	3	4	5	6	7	8	9	10
2g.	Right now: I feel Frustrated	FRUST	1	2	3	4	5	6	7	8	9	10
2h.	Right now: I feel Irritable	IRRITBL	1	2	3	4	5	6	7	8	9	10
2i.	Right now: I feel Confident	CONFDNT	1	2	3	4	5	6	7	8	9	10

3. Now think about when you were using tobacco…

		Not at All				Somewhat					Very Much
3a. How pleasurable was the <insert text responses to Q1>?	PLEASUR	1	2	3	4	5	6	7	8	9	10
3b. How satisfying was the <insert text responses to Q1>?	SATISFY	1	2	3	4	5	6	7	8	9	10

4. Right now: What are you doing (check all that apply)?
Checkbox with each category treated as a dichotomous variable with checked = 2 if unchecked = 1

Hanging out	___ HANG	Alcohol use	___ ALC
TV/Music/Movie	___ TVMUS	Marijuana use	___ POT
Telephone	___ PHONE	Party	___ PARTY
Texting	___ TEXTING	Physical activity/Sports	___ PHYS
Computer	___ COMPINT	Walking	___ WALKING
Reading	___ READING	Organized club	___ CLUB
Class work	___ SCHL	Transit/Driving	___ TRANS
Job	___ JOB	Nothing	___ NOTH
Eat/Drink	___ EATDRNK	Other	___ OTHER

5. When using tobacco: Were you with others? COMP

___1 Alone ___2 Alone-others nearby ___3 With other(s)

Checkbox with each category treated as a dichotomous variable with checked = 2 if unchecked = 1, NA= 0

5a. *If COMP = 2 (ALO = Alone-others nearby) or if COMP = 3 (WTH = With other(s)) then:*

5a1. Who was nearby (check all that apply)?

Family ___	ALOFAM/WTHFAM	Friends/ Peers ___	ALOFR/WTHFR
Romantic Partner/Spouse ___	ALOPART/WTHPART	Other ___	ALOOTH/WTHOTH
Coworkers ___	ALOWORK/WTHWORK		

5a2. *If 'Family' = "yes"* [ALOFAM/WTHFAM = 2] *then:*

Family nearby (With Family): Which family members were nearby (you with) (check all that apply)?

Mother	___ ALOMOM/WTHMOM	Father	___ ALODAD/WTHDAD
Brother(s)/Sister(s)	___ ALOSIB/WTHSIB	Other Relative	___ ALOREL/WTHREL
Children	___ ALOCHILD/WTHCHILD		

6d. *If COMP = 2 (ALO = Alone-others nearby) or if COMP = 3 (WTH = With other(s)) then:*

6d1. Are any of these people using any of the following tobacco products (check all that apply)?

Checkbox with each category treated as a dichotomous variable with checked = 2 if unchecked = 1, NA= 0

Cigarette	___ OCIG	Hookah	___ OHOOKAH
Little Cigar	___ OLCIGAR	Chew	___ OCHEW
Cigarillo	___ OCIGRILO	Dip/Snuff	___ ODIPSNF
Big Cigar	___ OBCIGAR	e-cigarette	___ OECIG
Snus	___ OSNUS	No one was using a tobacco product	___ ONONE
Vaporizer	___ OVAP		

7. **Right now: where are you?** LOCAT3

Your home	___1
Parent's home	___2
Friend's home	___3
School	___4
Work	___5
In car	___6
Other transit	___7
Bar/club	___8
Coffee shop	___9
Restaurant	___10
Hookah lounge	___11
Sport venue	___12
Entertainment venue	___13
Outside-public property	___14
Other	___15

8. Is smoking cigarettes allowed in this location? SMKLOCAT

Yes ___2 No ___1

9. In the last hour, have you seen or heard any anti-smoking or stop-smoking messages? ANTIMSG

Yes ___2 No ___1

10. In the last hour, have you seen or heard any cigarette or tobacco promotions or advertising? PROMSG

Yes ___2 No ___1

11. Did you drink alcohol in the last hour? ALCHOUR

Yes ___2 No ___1

12. Have you used marijuana in the last hour? POTHOUR

Yes ___2 No ___1

13. Now think about the time <u>just before</u> you used tobacco…

			Not at All				Somewhat					Very Much
13a.	Before Using Tobacco: I felt Happy	BHAPPY	1	2	3	4	5	6	7	8	9	10
13b.	Before Using Tobacco: I felt Relaxed	BRELAX	1	2	3	4	5	6	7	8	9	10
13c.	Before Using Tobacco: I felt Sad	BSAD	1	2	3	4	5	6	7	8	9	10
13d.	Before Using Tobacco: I felt Stressed	BSTRESS	1	2	3	4	5	6	7	8	9	10
13e.	Before Using Tobacco: I felt Angry	BANGRY	1	2	3	4	5	6	7	8	9	10
13f.	Before Using Tobacco: I felt Cheerful	BCHEER	1	2	3	4	5	6	7	8	9	10
13g.	Before Using Tobacco: I felt Frustrated	BFRUST	1	2	3	4	5	6	7	8	9	10
13h.	Before Using Tobacco: I felt Irritable	BIRRITBL	1	2	3	4	5	6	7	8	9	10
13i.	Before Using Tobacco: I felt Confident	BCONFDNT	1	2	3	4	5	6	7	8	9	10

6.C Appendix C Questions for the Random Prompt Interview Survey

The set of questions listed below were extracted directly from the FDA Communications Ecological Momentary Assessment Interviews Codebook (Appendix F.1). This document was authored by members of the research group: Dr. Robin Mermelstein and John O'Keefe. This particular document is an altered form of a previous document which had been used by the research group for over 10 years in similar tobacco use studies—including the PalmPilot study mentioned above. Outlined below are the sets of questions featured in the **Random Prompt Interview** within the E-cigarette Survey Application.

1. Think about how you feel right now…

			Not at All			Somewhat					Very Much
1a.	Right now: I feel Happy	HAPPY	1	2	3	4	5	6	7	8 9	10
1b.	Right now: I feel Relaxed	RELAX	1	2	3	4	5	6	7	8 9	10
1c.	Right now: I feel Sad	SAD	1	2	3	4	5	6	7	8 9	10
1d.	Right now: I feel Stressed	STRESS	1	2	3	4	5	6	7	8 9	10
1e.	Right now: I feel Angry	ANGRY	1	2	3	4	5	6	7	8 9	10
1f.	Right now: I feel Cheerful	CHEER	1	2	3	4	5	6	7	8 9	10
1g.	Right now: I feel Frustrated	FRUST	1	2	3	4	5	6	7	8 9	10
1h.	Right now: I feel Irritable	IRRITBL	1	2	3	4	5	6	7	8 9	10
1i.	Right now: I feel Confident	CONFDNT	1	2	3	4	5	6	7	8 9	10

2. Right Now: Do you have an urge to have/use any of the following tobacco products (check all that apply)?
Checkbox with each category treated as a dichotomous variable with checked = 2 if unchecked = 1

Little Cigar	___ ULCIGAR	Hookah	___ UHOOKAH
Cigarillo	___ UCIGRILO	Chew	___ UCHEW
Big Cigar	___ UBCIGAR	Dip/Snuff	___ UDIPSNF
e-cigarette	___ UECIG	Snus	___ USNUS
Vaporizer	___ UVAP	No urge to use any of these tobacco products	___ UNOURGE

3. Right now: What are you doing (check all that apply)?
Checkbox with each category treated as a dichotomous variable with checked = 2 if unchecked = 1

Hanging out	___ HANG	Alcohol use	___ ALC
TV/Music/Movie	___ TVMUS	Marijuana use	___ POT
Telephone	___ PHONE	Party	___ PARTY
Texting	___ TEXTING	Physical activity/Sports	___ PHYS
Computer	___ COMPINT	Walking	___ WALKING
Reading	___ READING	Organized club	___ CLUB
Class work	___ SCHL	Transit/Driving	___ TRANS
Job	___ JOB	Nothing	___ NOTH
Eat/Drink	___ EATDRNK	Other	___ OTHER

4. Are you with others? COMP

___ 1 Alone ___ 2 Alone-others nearby ___ 3 With other(s)

Checkbox with each category treated as a dichotomous variable with checked = 2 if unchecked = 1, NA = 0

5a. *If* COMP = 2 (ALO = *Alone-others nearby) or if* COMP = 3 (WTH = *With other(s)) then:*
'Who is nearby' or 'Who are you with' (check all that apply)?

Family ___ ALOFAM/WTHFAM	Friends/ Peers ___ ALOFR/WTHFR
Romantic Partner/Spouse ___ ALOPART/WTHPART	Other ___ ALOOTH/WTHOTH
Coworkers ___ ALOWORK/WTHWORK	

5a1. *If 'Family' = "yes"* [ALOFAM/WTHFAM = 2] *then:*

Family nearby /With Family: Which family members are nearby or with you (check all that apply)?

Mother	___ ALOMOM/WTHMOM	Father	___ ALODAD/WTHDAD
Brother(s)/Sister(s)	___ ALOSIB/WTHSIB	Other Relative	___ ALOREL/WTHREL
Children	___ ALOCHILD/WTHCHILD		

5b. *If* COMP = 2 (ALO = *Alone-others nearby) or if* COMP = 3 (WTH = *With other(s)) then:*

5b1. Are any of these people using any of the following tobacco products (check all that apply)?
Checkbox with each category treated as a dichotomous variable with checked = 2 if unchecked = 1, NA= 0

Cigarette	___ OCIG	Hookah	___ OHOOKAH
Little Cigar	___ OLCIGAR	Chew	___ OCHEW
Cigarillo	___ OCIGRILO	Dip/Snuff	___ ODIPSNF
Big Cigar	___ OBCIGAR	Snus	___ OSNUS
e-cigarette	___ OECIG	No one was using	
Vaporizer	___ OVAP	a tobacco product	___ ONONE

6. **Right now: where are you?** LOCAT3

Your home	___1
Parent's home	___2
Friend's home	___3
School	___4
Work	___5
In car	___6
Other transit	___7
Bar/club	___8
Coffee shop	___9
Restaurant	___10
Hookah lounge	___11
Sport venue	___12
Entertainment venue	___13
Outside-public property	___14
Other	___15

7. Is smoking cigarettes allowed in this location? SMKLOCAT
Yes ___2 No ___1

8. In the last hour, have you seen or heard any anti-smoking or stop-smoking messages? ANTIMSG
Yes ___2 No ___1

9. In the last hour, have you seen or heard any cigarette or tobacco promotions or advertising? PROMSG
Yes ___2 No ___1

10. Did you drink alcohol in the last hour? ALCHOUR
Yes ___2 No ___1

11. Have you used marijuana in the last hour? POTHOUR
Yes ___2 No ___1

6.D Appendix D Example Data Output for the Tobacco Use Interview

Below are some example data outputs—both encrypted and decrypted—from the Tobacco Use Interview. Please note that some of the questions have an answered value of 0, this means that the previous choices of questions did not lead the user to encountering this question (Figure 7).

Tob.tob (Encrypted):

```
tgubzn'! )/&%/'%&)_&:&%:'*! -! %! .! (! .! -! (! .! )! -! .! (! -! &! &! '! &! '! &! &! &! &! &! &! &! &!
'! &! '! &! &! &! %! %! %! %! %! %! %! %! %! %! %! %! %! %! %! %! %! %! %! %! %! &)! &! '!
&! &! '! .! (! .! (! -! .! -! (! .
```

userID.tob (Decrypted):

```
userID, 4/10/2014_1:10:25, 8, 0, 9, 3, 9, 8, 3, 9, 4, 8, 9, 3, 8, 1, 1, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 2,
1, 2, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 14, 1, 2, 1, 1, 2, 9, 3, 9, 3, 8, 9, 8,
3, 9
```

6.E Appendix E Example Data Output for the Random Prompt Interview

Below are some example data outputs—both encrypted and decrypted—from the Random Prompt Interview. Please note that some of the questions have an answered value of 0, this means that the previous choices of questions did not lead the user to encountering this question (Figure 7).

Rpt.rpt (Encrypted):

```
tgubzn'! )/&%/'%&)_&:)*:'(! .! (! .! .! )! .! (! -! .! &! &! '! &! &! &! &! &! &! &! &! '! &! &! '! &! &!
&! &! &! '! &! &! &! &! &! &! &!
&! %! %! %! %! %! %! %! %! %! %! %! %! %! %! %! %! %! %! %! %! %! %! &&! '! &! '! '! &
```

userID.rpt (Decrypted):

```
userID, 4/10/2014_1:45:23, 9, 3, 9, 9, 4, 9, 3, 8, 9, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 2, 1, 1, 1,
1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 11, 2, 1, 2, 2, 6.
```

6.F Appendix F External Documents:

The document cited below was referenced throughout the development of the E-cigarette Survey Prompt application. This document is not available online and so it has been cited within a separate appendix.

[1]Robin Mermelstein, John O'Keefe, "FDA Communications Ecological Momentary Assessment Interviews Codebook", External document, pp. 1-7, 2013.