****

**Lee Virtual School Mobile Application**

**CEN 4072 Test Report**

**Shelby Smith and Derek Bailey**

**April 25th, 2018**

**CEN 4935 Senior Project - Spring 2018**

**Instructors: Dr. F. Gonzalez & Dr. J. Zalewski**

**Department of Software Engineering**

**Florida Gulf Coast University**

**Ft. Myers, FL 33965**

**Copyright © 2018 by Florida Gulf Coast University**

**1. Testing Environment**

Testing will be conducted on a Windows 10 operating system with Visual Studio Enterprise 2017 and a MacOS 10.13 High Sierra operating system with Visual Studio for Mac 2017.

**1.1 Automatic Testing**

All automatic testing will be performed on the Windows 10 operating system with Visual Studio Enterprise 2017 installed. The NuGet packages of Nunit3 3.10.1 and Nunit3TestAdapter 3.10 will be installed, as well as AxoCover 1.1.5.20. The installation sites and instructions can be found below:

[**https://www.nuget.org/packages/nunit/**](https://www.nuget.org/packages/nunit/)

[**https://www.nuget.org/packages/NUnit3TestAdapter/**](https://www.nuget.org/packages/NUnit3TestAdapter/)

[**https://marketplace.visualstudio.com/items?itemName=axodox1.AxoCover**](https://marketplace.visualstudio.com/items?itemName=axodox1.AxoCover)

The software to be tested will be the user interface and the input validation for the Lee Virtual School Mobile Application developed by Shelby Smith and Derek Bailey. The app serves as an overall hub for resources for the virtual school including a calendar, webviews, and a native application.

**1.2 Manual Testing**

Manual testing will be performed on both the Windows 10 and MacOS 10.13 operating systems. A written test case will be authored to be conducted on both systems with correct input or UI navigation given, as well as expected output. Visual confirmation of the actual output versus the expected output will be the results of the test case.

**2. Test Results**

1. The software shall function as a mobile app for iOS and Android devices.

* Android 4.0.3 (Ice Cream Sandwich) or higher
* iOS 8.0 or higher

**Test Case 1.1** - UI Load test for Android

Inputs: No Inputs are required for this test case

Expected Output: Application successfully loads on the Android device.

Actual Output: Successful - Output matches expected output.

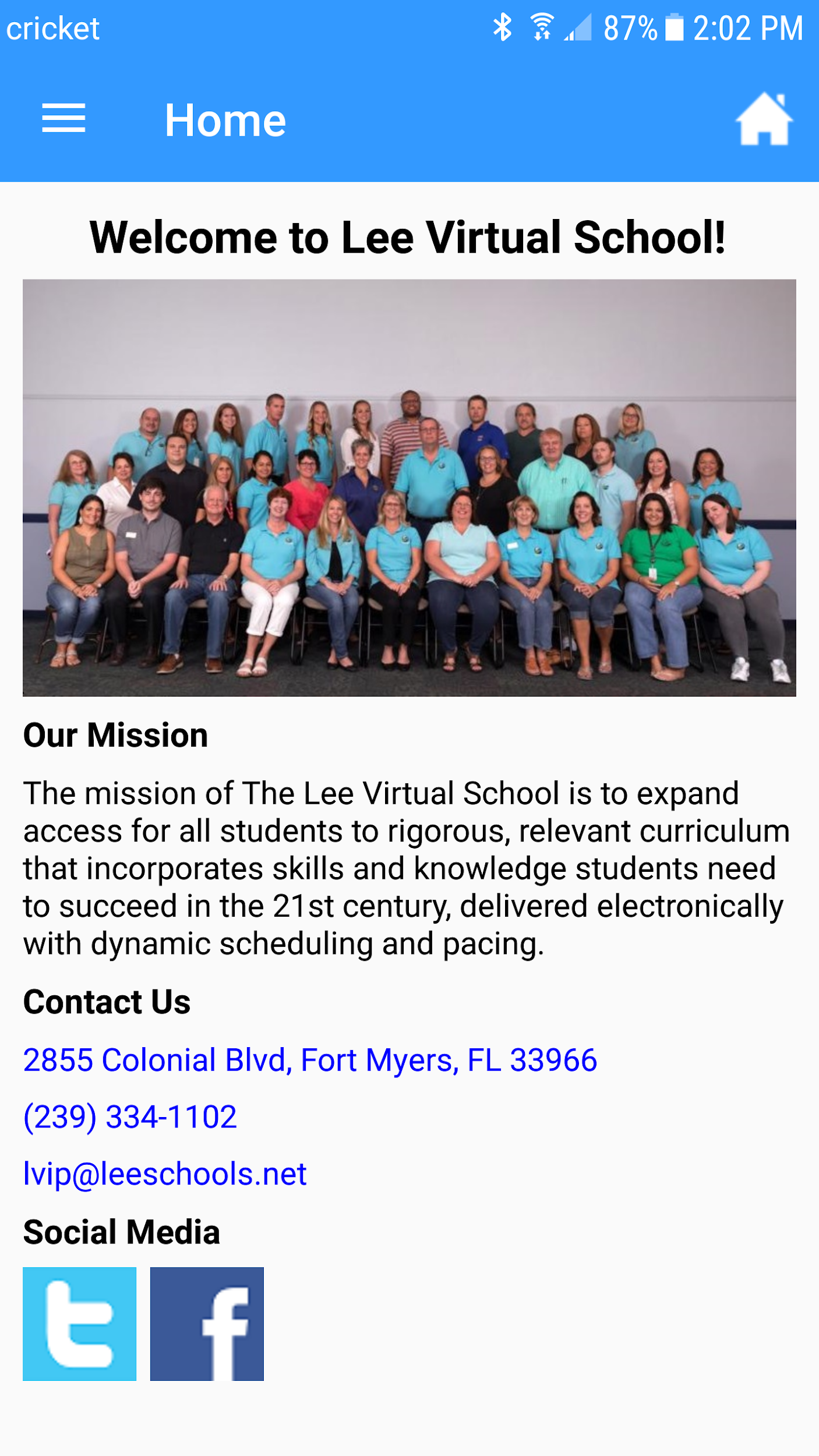
Discrepancies: None

**Test Case 1.2** - UI Load test for iOS

Inputs: No Inputs are required for this test case

Expected Output: Application successfully loads on the iOS device.

Actual Output: Successful - Output matches expected output.

Discrepancies: None

**Figure 2.1** Screenshot of the Home Screen of the App after it loads on the device.

2. The software shall implement a way to view events from a Google Calendar within the application.

* The software shall connect to the school’s Google calendar.
* The software shall pull information on events from the school’s Google calendar.

**Test Case 2.1** - Google Calendar Connection

Inputs: No inputs are required for this test case.

Expected Output: A successful connection to the Google calendar

Actual Output: Successful - Output matches expected output.

Discrepancies: None

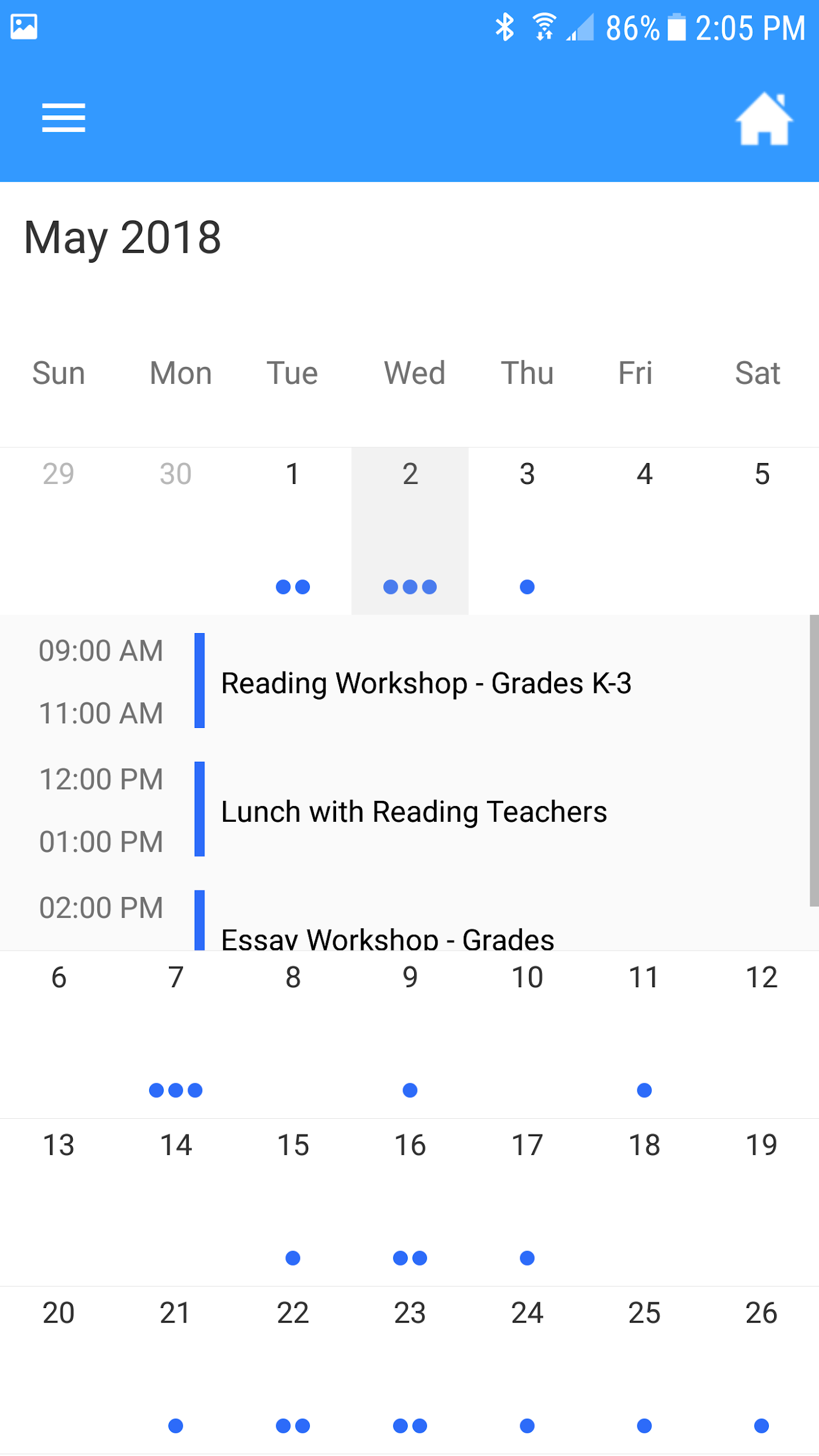
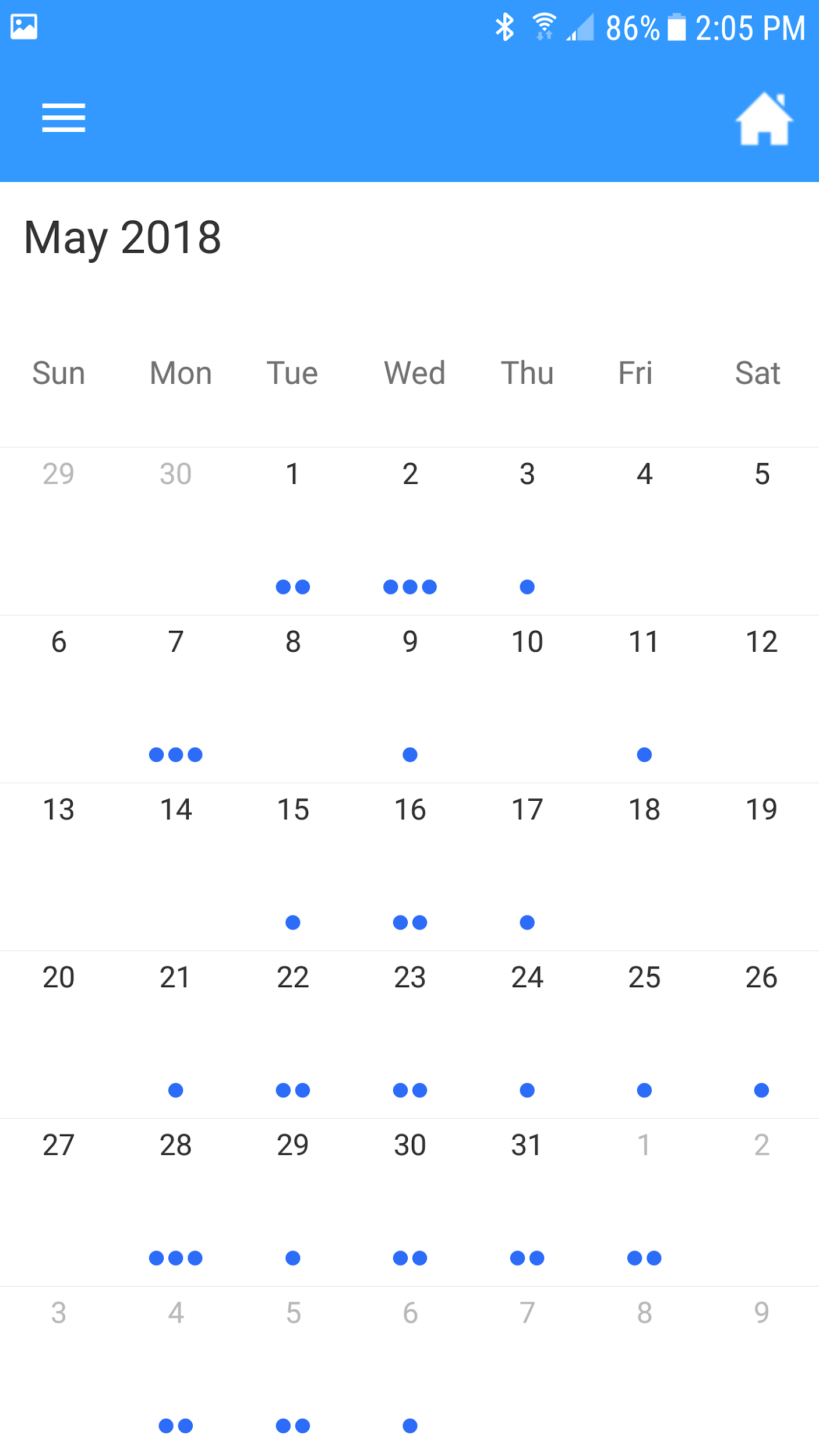
**Test Case 2.2** - Event Pulled from Calendar

Input: User clicks the calendar date to view the information

Expected Output: The UI will display the information for the event.

Actual Output: Successful - Output matches expected output.

Discrepancies: None



**Figure 2.2** Screenshot of the Calendar (left) and a date selected (right).

3. The software may optionally implement a way for a user to add an event from the school’s Google Calendar into their personal Google Calendar.

* The software shall connect to the school’s Google calendar.
* The software shall send the appropriate event information to the user’s personal Google calendar.

**Test Case 3.1** - Google Calendar Connection

Inputs: No inputs are required for this test case.

Expected Output: A successful connection to the Google calendar

Actual Output: Successful - Output matches expected output.

Discrepancies: None

**Test Case 3.2** - Send Event Information to personal Google Calendar

Input: User clicks the button to add an event to their personal Google Calendar

Expected Output: The app will open their native browser to add the event in their own Google Calendar after logging in.

Actual Output: Successful - Output matches expected output.

Discrepancies: None

4. The software shall provide a link for a student to login for their daily attendance on the District’s website.

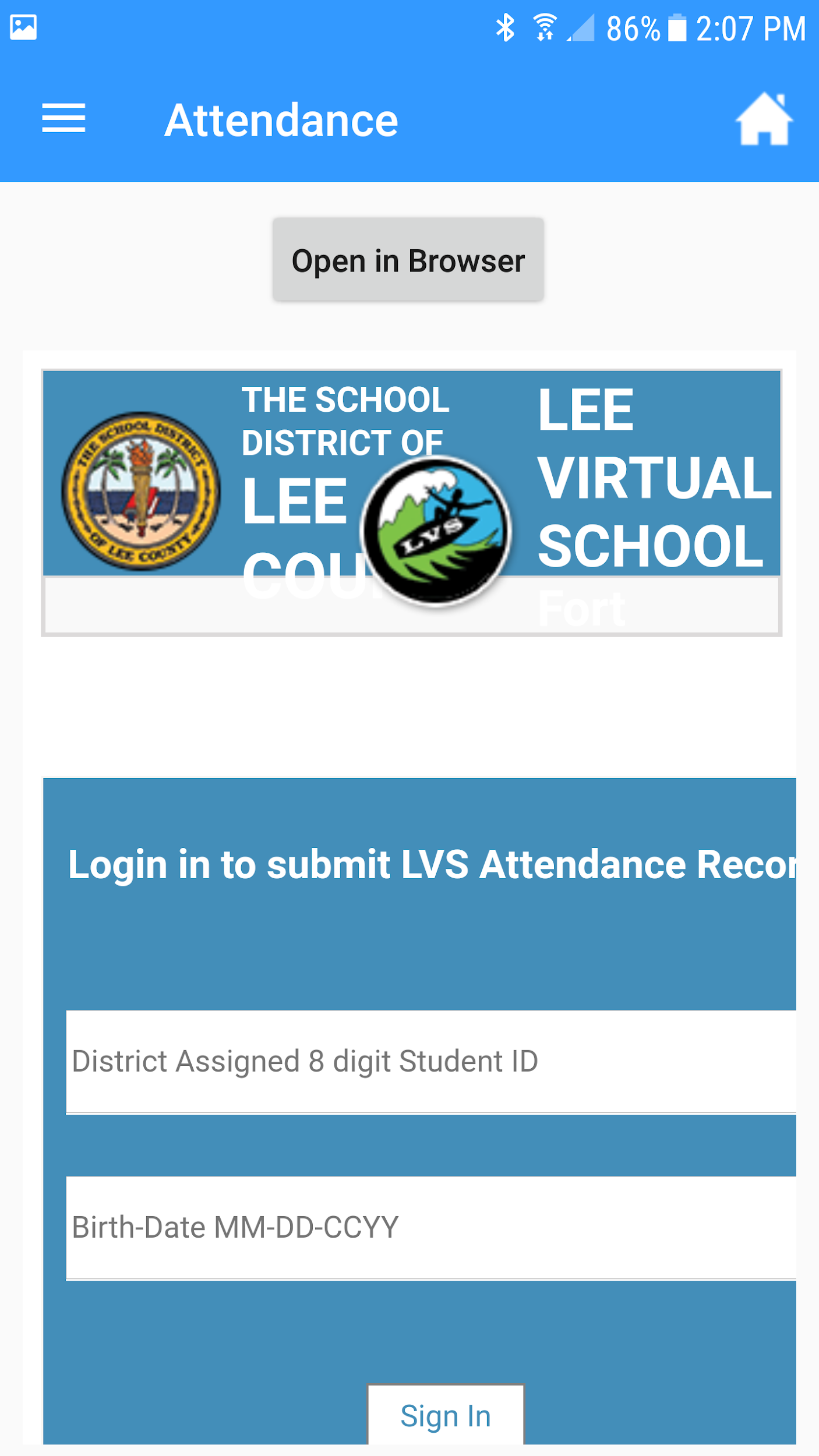
**Test Case 4.1** - Loading the Attendance Page

Input: User clicks the Attendance Menu Item

Expected Output: The Attendance Page loads properly into the WebView.

Actual Output: Successful - Output matches expected output.

Discrepancies: None



**Figure 2.3** Screenshot of the Attendance WebView.

5. The software shall provide the school’s staff contact information including (but not limited to) Email, Phone, Available Hours, Courses, and a link to the Genbook appointment scheduling website.

* The software shall retrieve and read the file with the contact information.
* The software shall populate the user interface with the file’s data.

**Test Case 5.1** - Google Sheets Connection and Data Retrieval

Input: User clicks the teachers menu item followed by a submenu item

Expected output: The Google Sheet will be successfully reached and read

Actual Output: Successful - Output matches expected output.

Discrepancies: None

**Test Case 5.2** - Populating data into the UI

Input: User clicks the teachers menu item followed by a submenu item

Expected output: The ListView will be populated with teacher names.

Actual Output: Successful - Output matches expected output.

Discrepancies: None

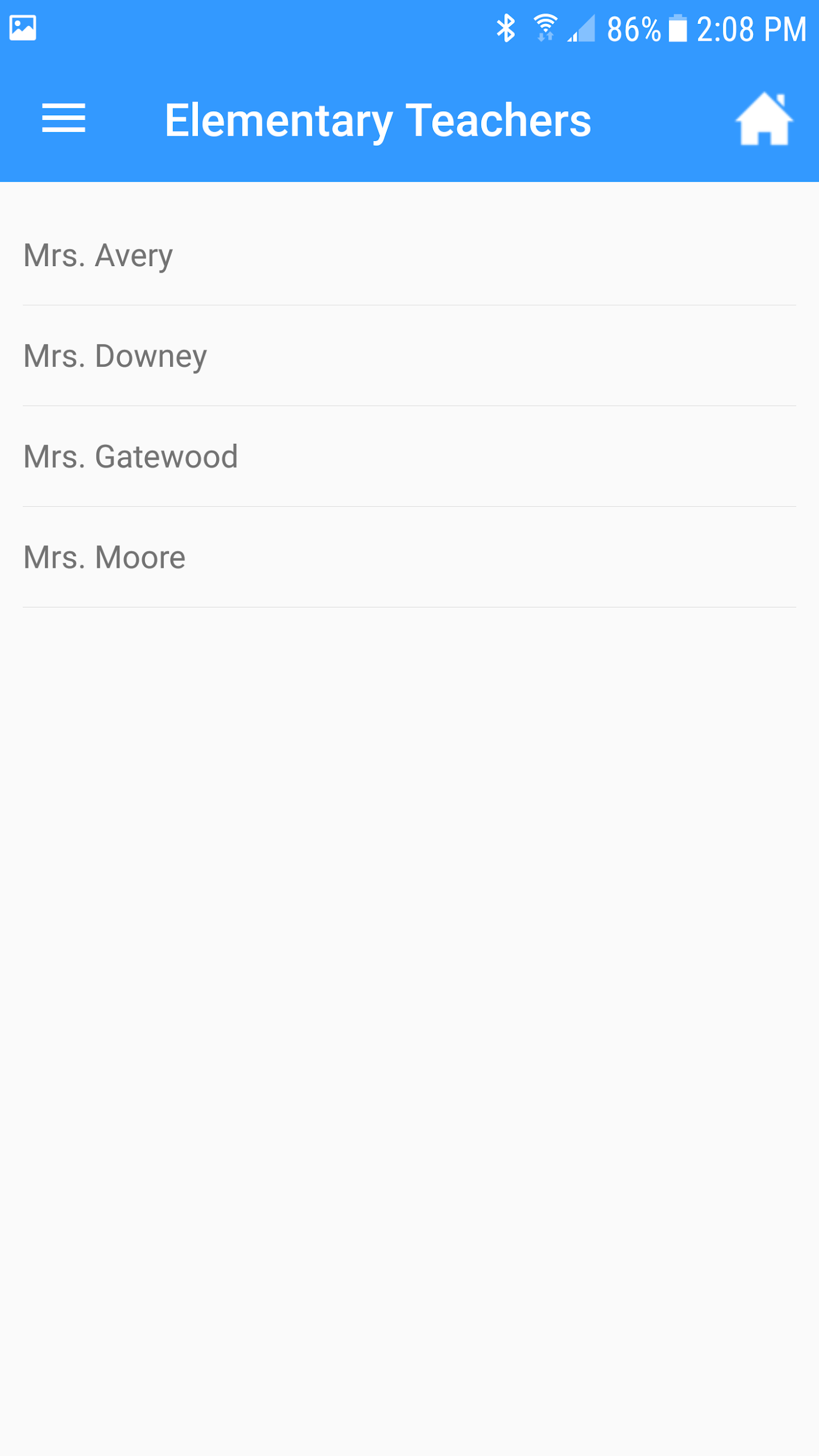
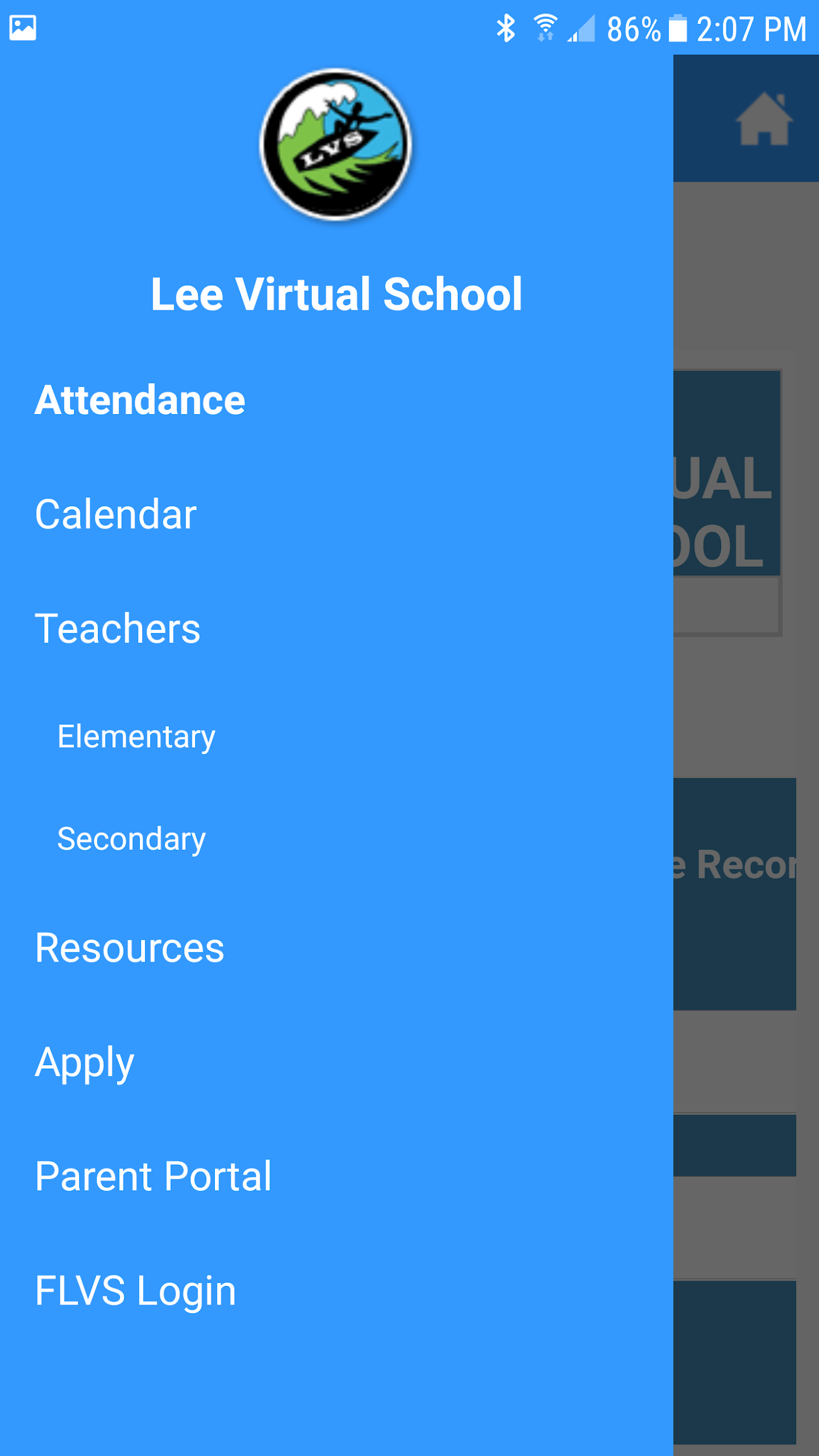
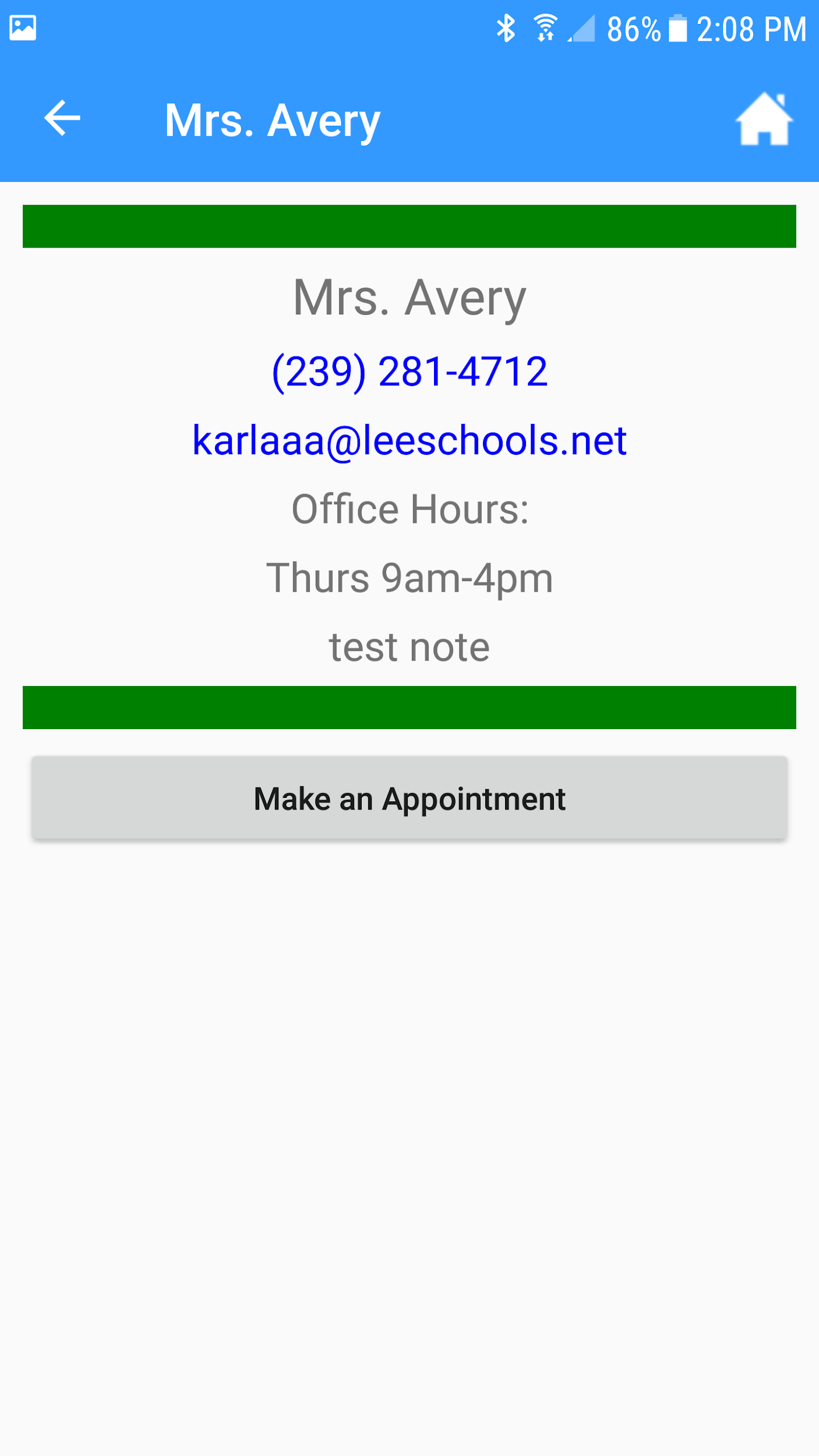
**Test Case 5.3** - Viewing More Information About the teacher

Input: User clicks a teacher’s name from the list

Expected Output: The UI will change to show all the information corresponding to the selected teacher.

Actual Output: Successful - Output matches expected output.

Discrepancies: None



**Figure 2.4** Screenshot of the teacher submenu (left), the list of teachers (middle), and the more information page for a selected teacher (right).

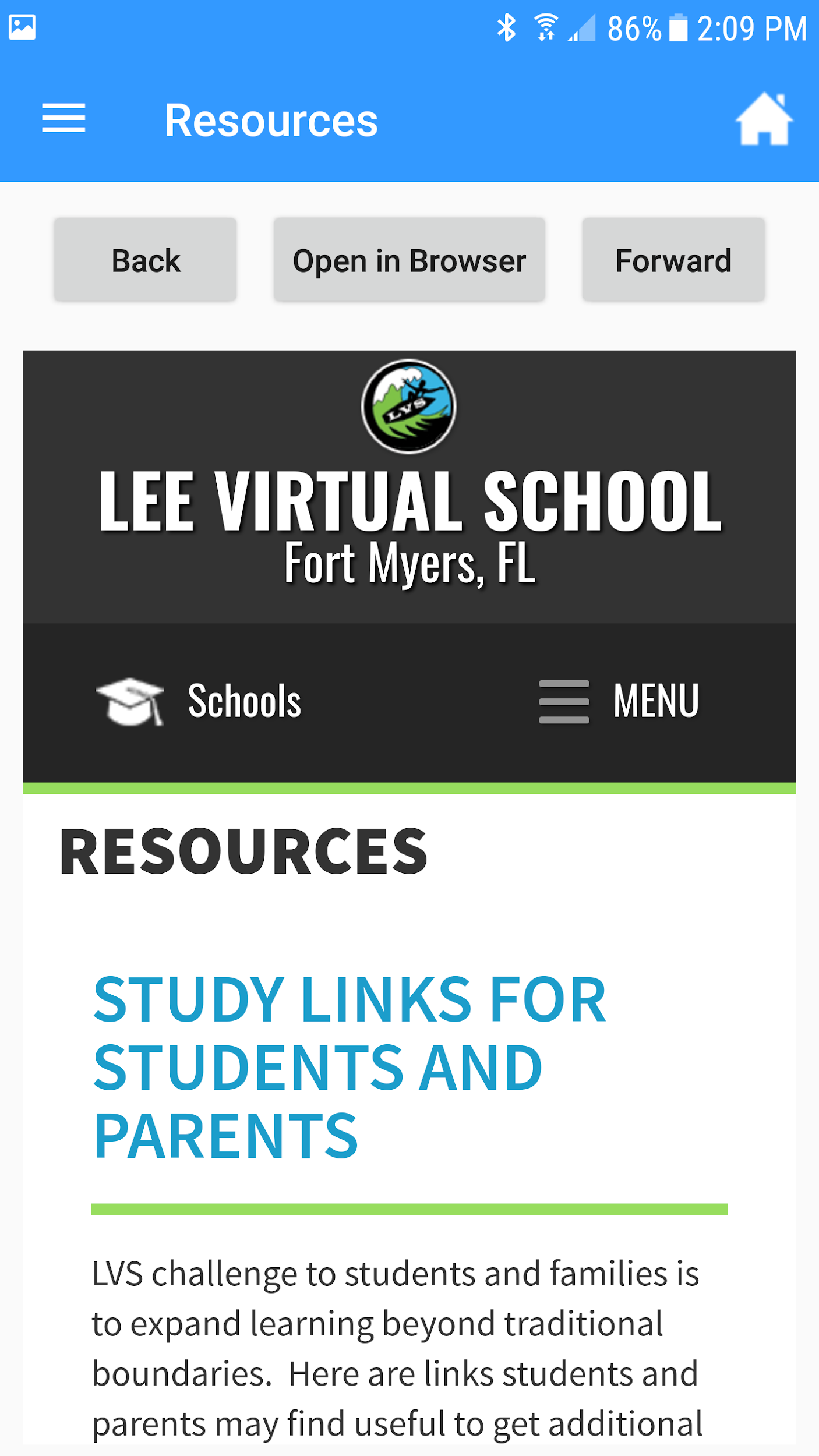
6. The software shall provide a list of resource links for students.

**Test Case 6.1** - Loading the Resource Page

Input: User clicks the Resource Menu Item

Expected Output: The Resource Page loads properly into the WebView.

Actual Output: Successful - Output matches expected output.

Discrepancies: None

**Figure 2.5** Screenshot of the Resources WebView

7. The software shall provide a “native” form to submit applications into Lee Virtual that includes all sections listed in the current PDF version of the application as shown in the Appendix LVS Application PDF.

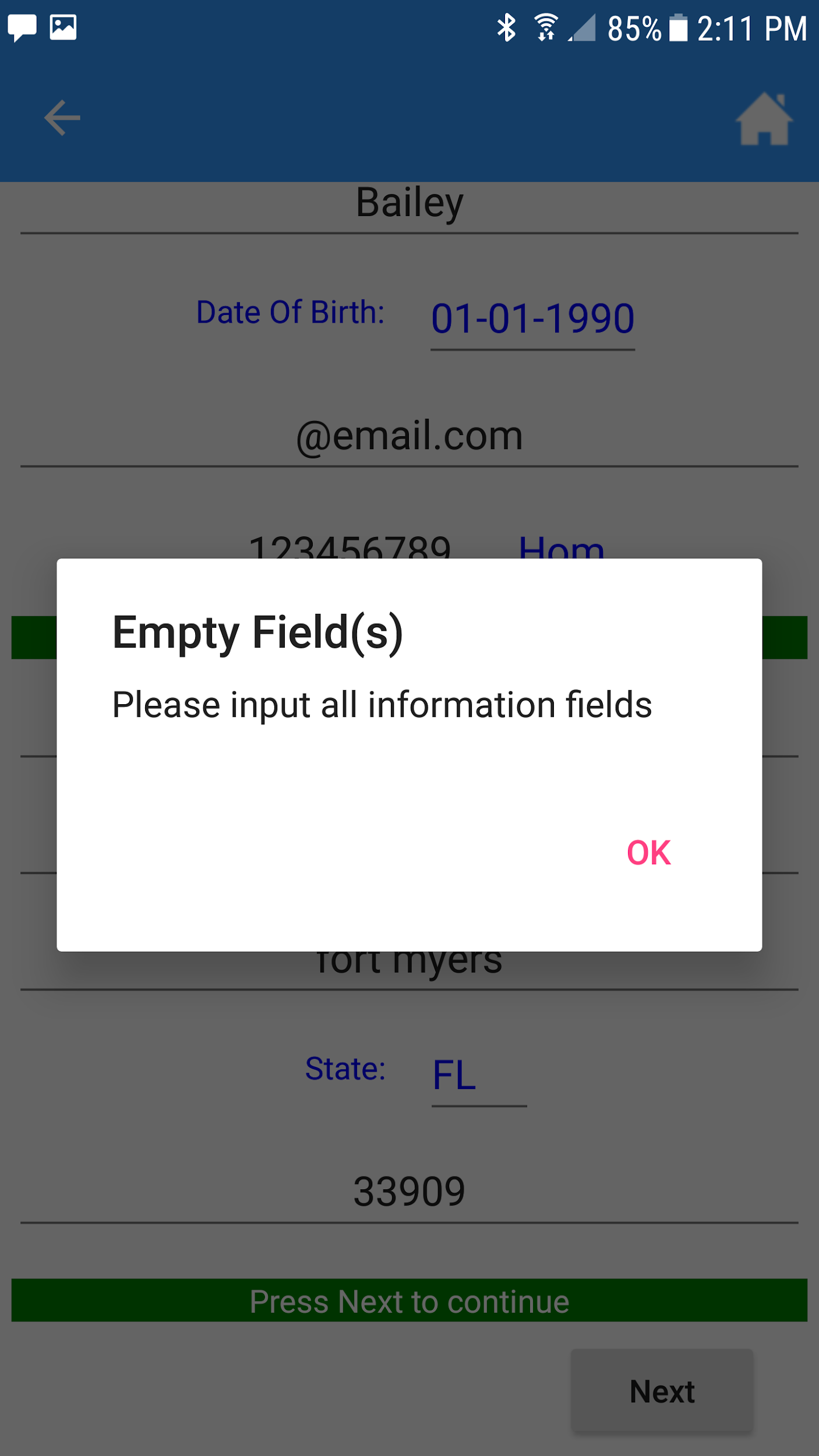
* The software shall validate all user input for correctness.

**Test Case 7.1** - Check For null or empty Entry fields

Input: User leaves the field empty

Expected Output: Alert Box telling the user to fill out all the fields

Actual Output: Successful - Output matches expected output.

Discrepancies: None

**Figure 2.6** Screenshot of the Empty/Null field Alert Box

**Test Case 7.2** - Check for correct email with nothing before the @ symbol

Input: “@email.com”

Expected Output: Alert Box telling the user the email is in an incorrect format.

Actual Output: Successful - Output matches expected output.

Discrepancies: None

**Test Case 7.3** - Check for correct email with an incorrect ending (no .com or .net etc.)

Input: “test@email”

Expected Output: Alert Box telling the user the email is in an incorrect format.

Actual Output: Successful - Output matches expected output.

Discrepancies: None

**Test Case 7.4** - Check for correct email with no @ symbol

Input: “testemail.com”

Expected Output: Alert Box telling the user the email is in an incorrect format.

Actual Output: Successful - Output matches expected output.

Discrepancies: None

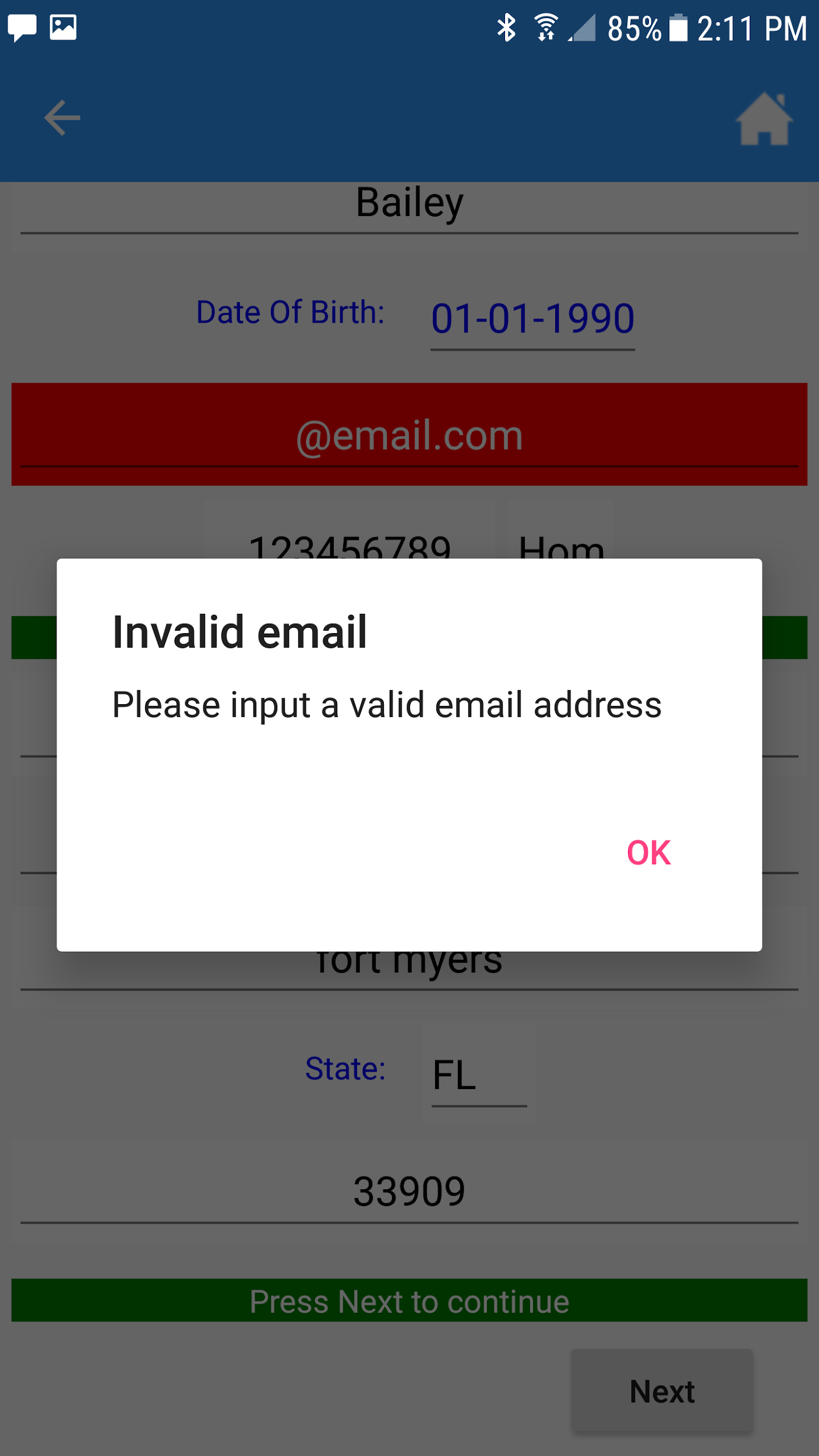
**Test Case 7.5** - Check for correct email with a correct email

Input: “test@email.com”

Expected Output: No alert will be shown and the program can proceed.

Actual Output: Successful - Output matches expected output.

Discrepancies: None

 **Figure 2.7** Screenshot of the Invalid Email Alert Box

**Test Case 7.6** - Check for valid Phone number with less than 10 digits

Input: “123456789”

Expected Output: An alert box telling the user to enter a correct phone number

Actual Output: Successful - Output matches expected output.

Discrepancies: None

**Test Case 7.7** - Check for valid Phone number with an alpha character

Input: “123456789F”

Expected Output: An alert box telling the user to enter a correct phone number

Actual Output: Successful - Output matches expected output.

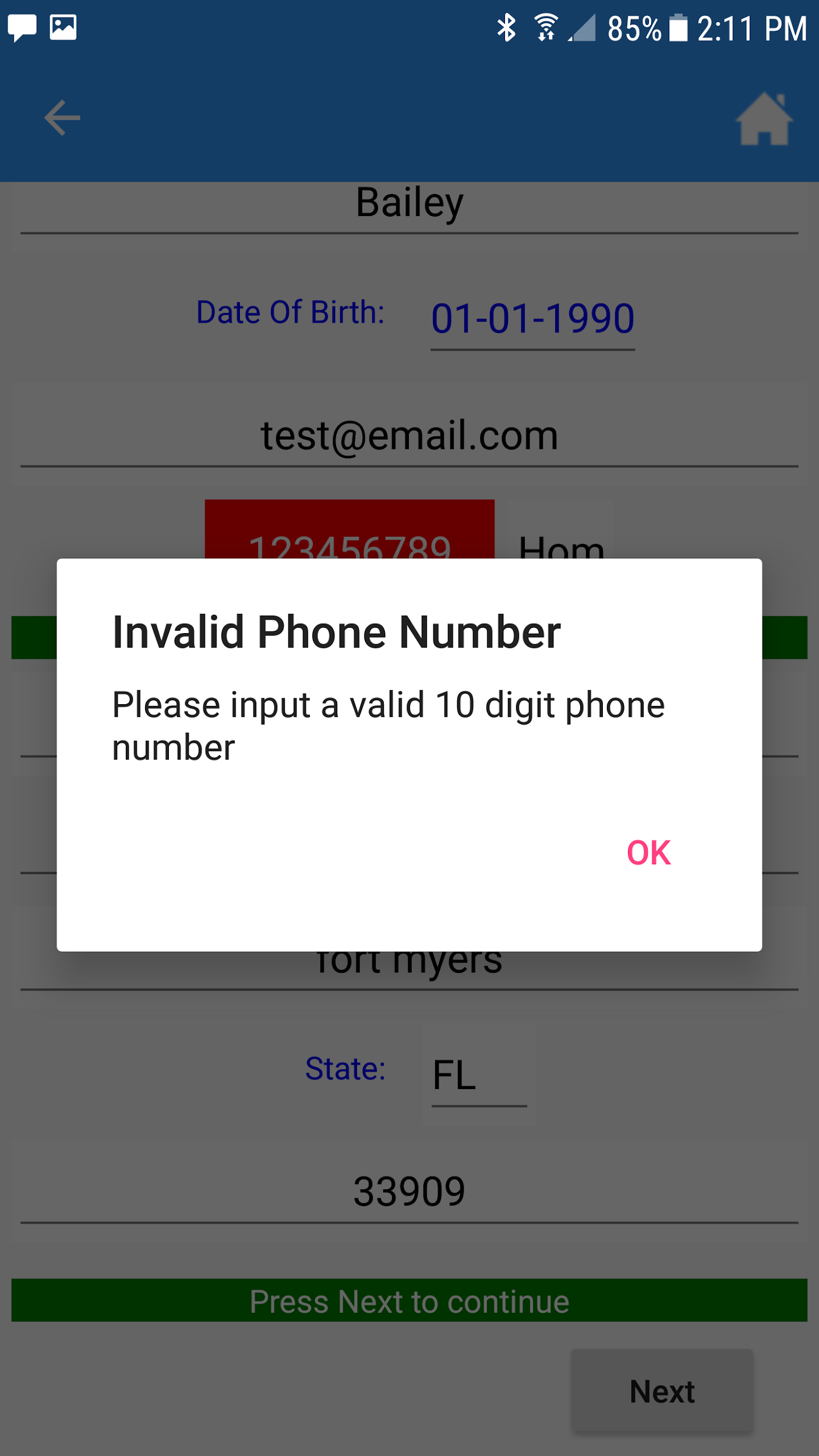
Discrepancies: None

**Test Case 7.8** - Check for valid Phone number with a valid phone number

Input: “1234567890”

Expected Output: No alert box is shown and the program can proceed.

Actual Output: Successful - Output matches expected output.

Discrepancies: None

**Figure 2.8** Screenshot of the Invalid Phone Number Alert Box

**Test Case 7.9** - Check for valid ZIP Code with less than 5 digits

Input: “1234”

Expected Output: An alert box telling the user to enter a correct ZIP code

Actual Output: Successful - Output matches expected output.

Discrepancies: None

**Test Case 7.10** - Check for valid ZIP Code with an alpha character

Input: “1234F”

Expected Output: An alert box telling the user to enter a correct ZIP code

Actual Output: Successful - Output matches expected output.

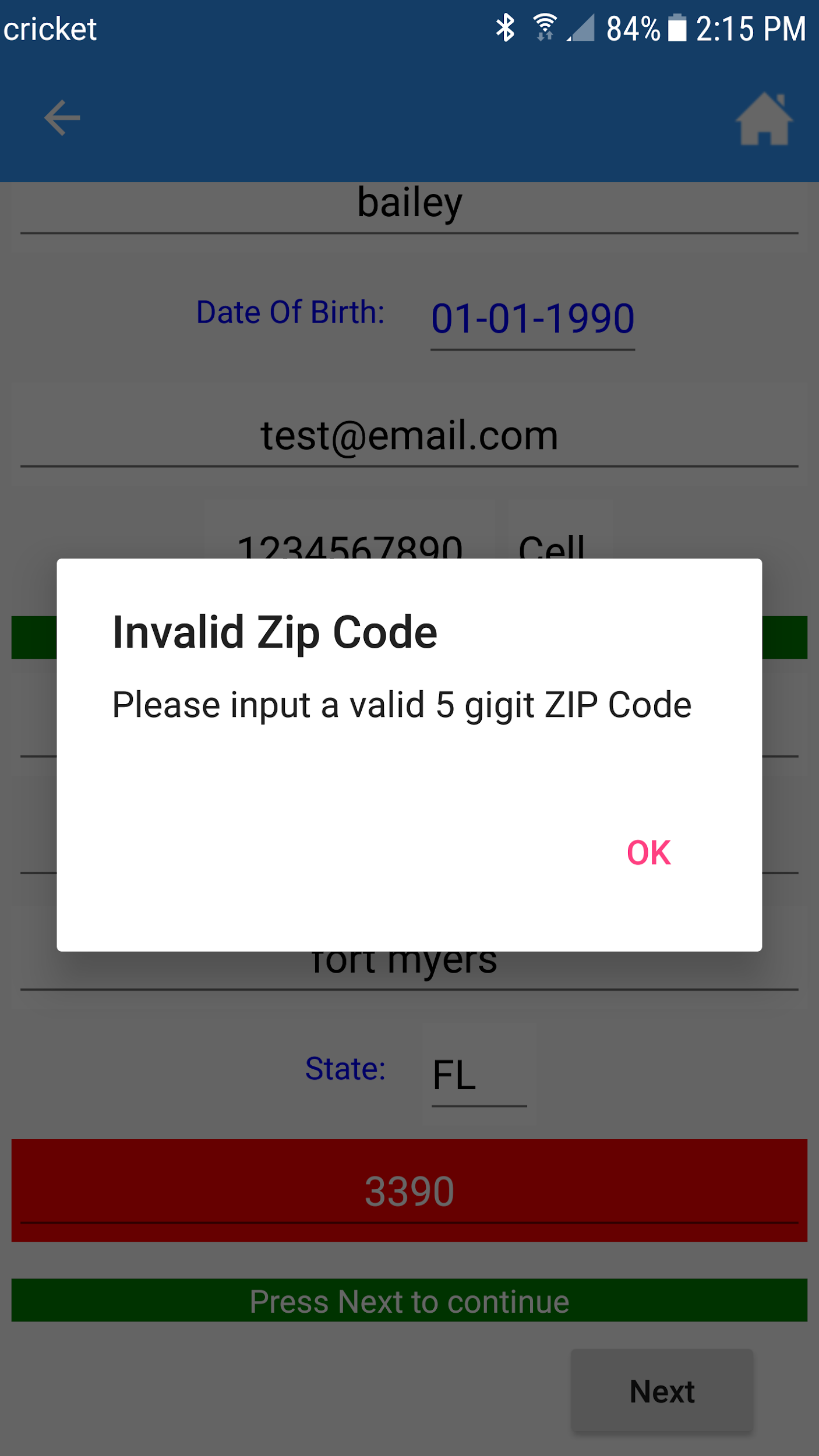
Discrepancies: None

**Test Case 7.11** - Check for valid ZIP Code a valid ZIP code

Input: “12345”

Expected Output: No alert box is shown and the program can proceed.

Actual Output: Successful - Output matches expected output.

Discrepancies: None

**Figure 2.9** Screenshot of the Invalid ZIP Code Alert Box

8. The software shall submit the application form to Lee Virtual School’s Applications Database via a RESTful web service upon completion of an application form by the user and requesting the submission.

* The software shall establish a connection with the LVS database.
* The software shall send data to the LVS database.

**Test Case 8.1** - Connection to LVS Database

Input: A connection request is sent to the LVS database server.

Expected output: A successful connection to the database.

Actual Output: Successful - Output matches expected output.

Discrepancies: None

**Test Case 8.2** - Send Data to the Database

Input: A JSON object will be sent to the database.

Expected Output: Successful transmission of the JSON object to the database.

Actual Output: Successful - Output matches expected output.

Discrepancies: None