1. PROJECT SCOPE

The client is in need to develop a Desktop, Android, iOS, and Windows application where the following will be included as the scope for the development.

1.1 DESKTOP APPLICATION

The application will be used mainly in the desktop which will connect the Android, iOS and the windows application.

The application is used to test the performance of the device (Android, iOS and windows) from the desktop. The desktop application will be having all the Android SDK, iOS SDK and the Windows SDK within it. The mobile device will be connected to the desktop via an USB cable and it will get detected, once the device is detected then the appropriate SDK will be used to connect with the mobile device. After the connection, for Android and Windows an APK will be sent which will be installed manually to the mobile device and for iOS the application will be installed from the test flight.

These mobile applications will not be uploaded to any of the stores.

After installation, the tests will be carried out and the result will be displayed on the desktop application. The following are various types of reports which will be available after the test.

Active Report

- Phone cards should be laid out in the pattern specified by the USB hub they are attached to if the hub has been configured.
- If a client has not been selected and the “start” button is pressed, it should show a modal dialog saying to please select a client.

Erasure Report

- Licenses should not be consumed when an erasure fails.
- Erasure types should be “factory reset” and “full erasure”. If “full erasure” is selected, and the device does not support it, the device should be marked failed.

Functionality Report

- Wireframes should be fairly explanatory.
- In the report options section, there should be a dropdown that shows the different test suites that can be selected.
- If a test suite is not selected before pressing “start”, it should show a modal dialog saying to please select a test suite.

Imaging Report

- Not shown in wireframes.
- Phone cards should have a dropdown selector added to them to select from a list of images.
- Device images should be automatically selected if the image has filters matching the device.
- If there are multiple images available it should select the most recent image.
- If there are no images with filters matching the device, the image selection dropdown should show the available images that have no filters associated.
- Licenses should not be consumed when writing an image fails.
- If any devices do not have an image selected, and a technician presses the “start” button, it should show a modal dialog with the text “Some devices do not have images selected. Start the report anyway?” With the options “no” (default), and “yes”

Multiple Step Report

- Not shown in wireframes.
- Should be a single report where you can step through functionality testing, imaging, and erasure.

After the report is generated it can be saved as PDF or XLSX. The reports can also be filtered. It would like to be able to filter on any associated sequelize models.

- `is` - Filter on the enumerable status column.
- `after` - Filter on reports created on or after the supplied date.
- `on` - Filter on reports created on the supplied day.
- `before` - Filter on reports created on or before the supplied date.

- `client` - Filter on reports associated with clients matching the supplied name.
- `technician` - Filter on reports associated with technicians matching the supplied name.
- `serial` - Filter on reports that have a device with the supplied serial number.
- `imei`- Filter on reports that have a device with the supplied IMEI.
- Logging should include devices as they are connected; erasure and imaging start/stop/abort/finalize; technician sign in/sign out; edits to technicians, device images, clients, sites, etc.
- Forward hardware error status codes from client device to the host machine and log them. If an error status code applies to an action initiated from the host machine (functionality tests, erasure, rebooting a client, imaging, etc), some relation should be made between that action and the logged error for later lookup.

1.2 ANDROID / iOS / WINDOWS APPLICATION

Functionality Tests

- If phone is iCloud/Google Account locked and the app cannot be installed: when running functionality report the phone should be marked automatically, with the option for a technician to mark that the app could not be installed for another reason.

Manual Tests

The application will be compatible for both mobile and tablet devices.

- These tests should be run as necessary provided a client device has the hardware available for testing. If a client device does not have the hardware required for a test then the test result should be marked “N/A”.
- Transitions between tests should be a horizontal slide from right to left over a 500ms duration using a quadratic ease-out function.
- Most manual functionality tests have one or two buttons at the bottom to allow the technician to mark the test as “passed” or “failed”. Tapping either button type should mark the test as indicated on the button, and move to the next test.
- Manual tests will typically have a 20 second delay before moving to the next test with the following exceptions:

    - Dead Pixels - When a technician presses the "start" button to begin the test, the countdown should pause until the screen finishes cycling between colors.
    - Any test that requires the technician to leave the app and return should pause the countdown.
- If a test is not passed within 20 seconds it should be marked as "timed out".
- If the touch screen tests fail, all subsequent tests should be marked as "untested", as a technician will not be able to mark them as failed.

Automated Tests

These tests should be run as necessary provided a client device has the hardware available for testing. If a client device does not have the hardware required for a test then the test result should be marked "N/A".

- WiFi - Connect to a wireless access point on the host machine.
- Cellular - Can connect to a cellular network.
- Bluetooth - Can establish a connection to the host machine with Bluetooth.
- GPS - Can receive a GPS signal, and pull a location (latitude, longitude, altitude) from the client.
- Device is not jailbroken - Check to see that the client device is jailbroken or rooted.
    - One method would be checking to see if Cydia is installed on iOS devices, but other methods should be used to ensure this test is accurate.
    - This should also check to see if an Android device is rooted.
- Battery charge
- Vibration + Accelerometer - Read accelerometer and then pulse the vibration at different rates while continuing to read values from the accelerometer. Compare values to determine if both the accelerometer and vibration are working.
- Camera (back) + Flashlight - Take a picture from the rear camera with the flashlight off, then turn the flashlight on and take another picture. Average the lightness of each image.
- Camera (front) Read QR - Read a test QR code from the front camera.
- Camera (back) Read QR - Read a test QR code from the back camera.
- Microphone + Speaker - Play a number of different tones from the speaker and record the output. Analyze the recording to see if the tones were played by the speaker. This test should run with the volume set to 25%, 50%, 75%, and 100%, stopping when tones

are detected and saving the volume level required before tones were detected with the report.

- USB Data - Verify that the device can transfer data over USB.
- USB Power - Verify the client device is receiving power over USB.
- IMEI Blacklist - If the device has an IMEI against global databases to see if the device's IMEI is blacklisted.
- SIM Card - See if a SIM card is present. The test should fail if no SIM card is present.
- Activation Lock - See if the client device is iCloud locked or Google Account locked.
- Carrier Lock - See if the client device carrier locked.
- Warranty - See if the client device is within warranty.

Note:

- Wireframes will be provided by the client
- This application will be supported only for the following version of the devices
    - Android: 2.3 and above
    - IOS: 5.0 and above
    - Windows: 8, 8.1 and 10

2. TECHNOLOGY STACK

2.1 TECHNOLOGY USED

Technology Stack	Platform
Desktop	Electron framework
Android and IOS	Cocoon JS
Windows	Native application
Database	SQLite
Design	Angular JS, ngMaterial