# Mobile App Documentation

## Set-Up Instructions:

The mobile app is simple to set up. When it is first installed and opened, it will show the user with a log in screen, shown below. The log in screen will prompt the user with a username, password, and an IP address of the server. The port number is required for the server as well or the app will not work.

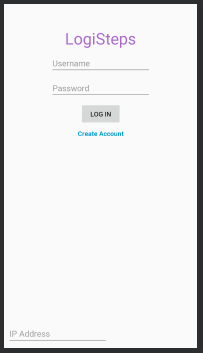


Figure 1: The log in screen for the mobile app

The user also has the option of creating an account by clicking the blue hyperlink. That will pop-up a screen that asks for more information, as shown below. All the information must be entered in order to successfully create an account. After either the log in screen, or the create account screen, the app will connect with the server and verify that the user is valid. If the user is valid the app moves to the main screen.

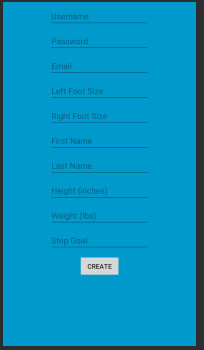


Figure 2: The create account screen for the mobile app

In the main screen, shown below, the user has a few different data entries shown to them. It shows the steps taken that day from the server, the user’s goal for the day, average steps per hour, and a projected for the day based on that information. The user also has the option of selecting one of the shoes, listed right foot and left foot as buttons, and connecting one of our LogiSteps insole devices to it. Once that is connected, the app will start receiving data and sending it to the server for further processing. Both must be done separately in order to get an accurate reading.

The user also has an account button. Here they can change accounts from one that is saved in the mobile device or simply log out.

The user has the ability to change the server IP and port as well. That is done by pressing the circular button between the right and left foot buttons and entering the information. Once this is done, the app then verifies that the server is there and that the user has an account there.

One thing that is not currently used but is in the app, is shoe type. It currently gives four options, however none of the other components currently have anything in their systems to handle this definition. The user is welcome to specify what type of shoe they have, but it will not be factored into their data.



Figure 3: The main screen for the mobile app

## Development:

### Beginning:

The app started as a shell. The GUI was designed first for the main screen as well as the shell for the two objects that would be used in the app; the foot connection and the server connection. Once the main screen was designed and created in android studio, all the buttons were tested and confirmed that the app did not crash. The idea for the shell was that it would be able to connect to two insoles and a server simultaneously as well as do some computing internally. It was also to be a smaller app in comparison to most android apps.

### Issues:

The biggest issue was the Bluetooth. The connection protocols and functions on the Android development website are out of date for our application. This took a few weeks to figure out. The issue was in the UUID of the device the app is trying to connect to. There is supposed to be a function that gets the UUID of the Bluetooth Device, however the function always returns null. It is unclear why this is, however, there is a workaround that can be done. This workaround brought up a few exceptions that could happen that needed to be checked, so it is not the most efficient, however it works.

The only other issue that was presented is needing all the other components to be complete to fully test the app. Since it is the mid-point, it needs the data from the device and ability to send it to the server tested. The server has been tested, but the device data has not been fully tested. The functions are all implemented, just not tested. Once all the other parts are functioning, the app will be fully tested.

### Where the app is now:

The app is currently in a crude version of its final form. As stated before, not everything has been fully tested, but it all has been partially tested. The server connection is almost completely tested as well as the internal processing functions. The Bluetooth is confirmed to pair successfully and the format of the data is known and is being parsed, however, the testing of passing data is not completely tested. As further versions come out of the app, it will run smoother and have a simpler UI for the users.

## Test Plan:

### Bluetooth Connection:

The first step in the mobile app is to get Bluetooth connecting. This is a main component of the mobile app and integrating it with the physical hardware of the LogiSteps system. Once this is completed, then when the main board of the device is completed, more testing can occur. For now, the connection is being established with a Windows 10 PC.

Initially, there is a problem with this method. The error that was coming up was one that made little sense. With a little internet searching, it is determined that the UUID of the device was the issue. All the documentation on the Android website said that the way the UUID is found originally is the correct way. However, in a deeper internet search, it is found that the documentation is out of date and a different method needed to be used. After much time and searching, the connection is established.

### Server Connection:

The next step is to get the server connected. Meeting up with a teammate and getting a local version of the server installed is the first part of this. After that is done, being able to connect from an external device is next. The documentation for the type of server it is provides that answer. It is done when starting the server by putting in 0.0.0.0:port at the end of the command. This links the server to the IP address on the device it is being run. Once the command is run, using a different device and inputting the IP of the server, followed by the port, followed by /logisteps, will allow that device connection to the server. This is, if they are on the same network.

### Server Interaction (Log in):

Since the connection is now set up, now interaction with needs to be tested. This needs to be done inside the app itself. The most basic way to interact with the server is to verify an account. The computer app Postman is used to create an account with the server. Then, in the mobile app in debugging mode, the request is made to the server to get the account. Luckily, Postman has a function that converts their requests into code that is useable for the app. This makes testing simple. Once the request is made, the debugger stops after the response is received and tells the user if there was a response and if so, what it is. If the response code is 200, everything went right.

### Device Interaction:

The next step is to test the interaction with the physical device. This is later in the test plan because it is waiting on the device to be ready. Once the device is ready there are two things to test. The first is the initial connection. Upon initial connection, the app must send the time in milliseconds from 1970. To test that, the board will send back an ack of some sort when it receives it.

The second part is getting data from the device. To test this, the app will be put in debug mode, and set to stop when it receives over Bluetooth. This will inform the developer of the format that the data is coming in and that data is coming in. Once the data is received and the developer knows the format of the data, the data can be parsed and used to update the UI and send to the server.

### Server Interaction (Post step):

The last thing to test in the mobile app is posting a step to the server. This will be tested similarly to the log in function in that the Postman helps create the request and the debugger is used to check the response. The difference is the code will be 201 if working. Once this is tested and the server shows 201, the mobile app will have all functionality. All that will be needed after this will be to shine it up and make it smoother.