Build a QA automation framework from scratch for the simple calculator iOS app. We'll be using the Appium framework, which is a popular choice for automating mobile applications.

To get started, make sure you have the following prerequisites:

Xcode: Install Xcode on your macOS machine.

Homebrew: Install Homebrew, a package manager for macOS.

Node.js: Install Node.js using Homebrew by running the command: brew install node.

Appium: Install Appium using Node.js by running the command: npm install -g appium.

Appium Doctor: Install Appium Doctor using Node.js by running the command: npm install -g appium-doctor.

Once you have the prerequisites set up, follow these steps:

Step 1: Set up your iOS environment

Launch the iOS Simulator from Xcode.

Ensure you have a valid iOS Simulator running.

Note down the simulator's UDID, which can be found by running the command: xcrun simctl list.

Step 2: Install Appium dependencies

Install the necessary Appium dependencies by running the command: appium-doctor --ios.

Step 3: Create a new directory for your automation project

Open a terminal and create a new directory for your automation project. Navigate to the directory using the cd command.

Step 4: Initialize a new Node.js project

Run the command: npm init -y. This will create a new package.json file.

Step 5: Install Appium and related libraries

Install the required libraries by running the following commands:

css

Copy code

npm install appium --save-dev

npm install appium-webdriverio --save-dev

npm install webdriverio --save-dev

Step 6: Set up the test framework

Create a new file called calculator.test.js in the project directory.

Step 7: Write the test cases

Open the calculator.test.js file and add the following code:

javascript

Copy code

const webdriverio = require('webdriverio');

const opts = {

path: '/wd/hub',

port: 4723,

capabilities: {

platformName: 'iOS',

platformVersion: 'YOUR\_IOS\_VERSION',

deviceName: 'YOUR\_DEVICE\_NAME',

app: 'PATH\_TO\_YOUR\_CALCULATOR\_APP',

automationName: 'XCUITest',

udid: 'YOUR\_SIMULATOR\_UDID',

},

};

(async () => {

const client = await webdriverio.remote(opts);

try {

await client.init();

// Test addition

const result1 = await client.$('~btn\_1').click();

const result2 = await client.$('~btn\_plus').click();

const result3 = await client.$('~btn\_2').click();

const result4 = await client.$('~btn\_equals').click();

// Test subtraction

const result5 = await client.$('~btn\_5').click();

const result6 = await client.$('~btn\_minus').click();

const result7 = await client.$('~btn\_3').click();

const result8 = await client.$('~btn\_equals').click();

// Test multiplication

const result9 = await client.$('~btn\_4').click();

const result10 = await client.$('~btn\_multiply').click();

const result11 = await client.$('~btn\_2').click();

const result12 = await client.$('~btn\_equals').click();

// Test division

const result13 = await client.$('~btn\_6').click();

const result14 = await client.$('~btn\_divide').