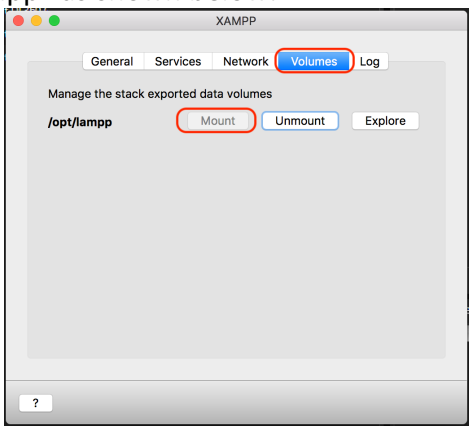
CoinAmount Application

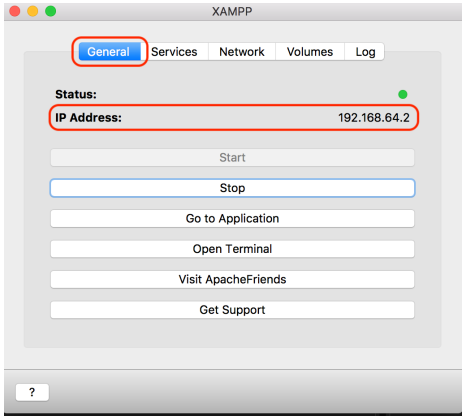
Installation

The application is developed using HTML, CSS, and JS. To run the application, you need to follow the following steps:

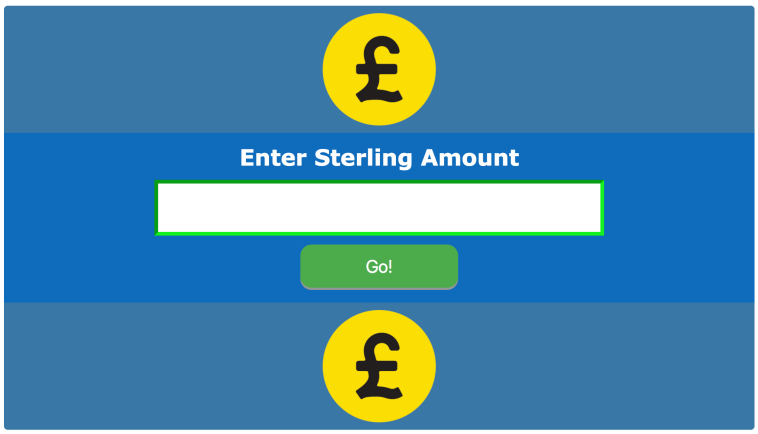
1. Download XAMPP for your operating system from the LINK.
2. Install XAMPP application and Start Apache service from Services tab.
3. Unzip the CoinsAmount.zip file. This should extract a folder called “CoinsAmount”
4. If you are on a windows machine: follow the following steps:
    a. Copy the folder “CoinsAmount” to the “htdocs” folder in the path: C:\xampp7\htdocs\
    b. Open your preferred web browser (Chrome, Firefox, Safari, IE) and go to the address: http://localhost/CoinsAmount or http://127.0.0.1/CoinsAmount
5. If you are using MAC OS:
    a. From XAMPP application panel, go to “Volumes” tab and press “Mount” to mount the “/opt/lampp” as shown below:



    b. From the same tab, press “Explore” button to open the root folder of the Apache server.
    c. Copy the folder “CoinsAmount” to the “htdocs” folder in the root Apache server.
    d. From XAMPP application panel, go to “General” Tab to get the IP address of your local Apache server as shown below:



    e. Open your preferred web browser (Chrome, Firefox, Safari, IE) and go to the address: http://192.168.64.2/CoinsAmount . Note that the used IP address is the same as the address from the previous step.
6. The following home page should appear and you should be able to start using the app.



Code Test Cases

I have used Qunit library to apply unit testing on the JavaScript code. I have created a HTML page called unitTesting that shows the tests results. All tests are implemented in the "js/tests.js" file.

The following code shows the implemented test cases:

1 Calc Coins function test: this function counts how many coin in the input value and returns it in the form of an array. All tests were successful.

```
QUnit.test("CalcCoins test", function(assert) {
  assert.deepEqual(CalcCoins("£12"), [6, 0, 0, 0, 0, 0, 0, 0]);
  assert.deepEqual(CalcCoins("123p"), [0, 1, 0, 1, 0, 0, 1, 1]);
  assert.deepEqual(CalcCoins("432"), [2, 0, 0, 1, 1, 0, 1, 0]);
  assert.deepEqual(CalcCoins("213p"), [1, 0, 0, 0, 1, 0, 1, 1]);
  assert.deepEqual(CalcCoins("£16.23p"), [8, 0, 0, 1, 0, 0, 1, 1]);
  assert.deepEqual(CalcCoins("£14"), [7, 0, 0, 0, 0, 0, 0, 0]);
  assert.deepEqual(CalcCoins("£54.04"), [27, 0, 0, 0, 0, 0, 2, 0]);
  assert.deepEqual(CalcCoins("£23.33333"), [11, 1, 0, 1, 1, 0, 1, 1]);
  assert.deepEqual(CalcCoins("001.41p"), [0, 1, 0, 2, 0, 0, 0, 1]);
  assert.deepEqual(CalcCoins("13x"), null);
  assert.deepEqual(CalcCoins("13p.02"), null);
  assert.deepEqual(CalcCoins("£p"), null);
});
```

2 Convert To Pennies function test: this function will check if the input is in the style of GBP or Pennies. If the value is in GBP style, it will be converted into Pennies and return it in its equivalent pennies amount. If the value is in Pennies style, it will be returned as it is. All tests were successful.

```
QUnit.test("Convert To Pennies test", function(assert) {
  assert.equal(convertToPennies("£12.34"), 1234);
  assert.equal(convertToPennies("123p"), 123);
  assert.equal(convertToPennies("432"), 432);
  assert.equal(convertToPennies("213p"), 213);
  assert.equal(convertToPennies("£16.23p"), 1623);
  assert.equal(convertToPennies("£14"), 1400);
  assert.equal(convertToPennies("£54.04"), 5404);
  assert.equal(convertToPennies("£23.33333"), 2333);
  assert.equal(convertToPennies("001.41p"), 141);
  assert.equal(convertToPennies("13x"), -1);
  assert.equal(convertToPennies("13p.02"), -1);
  assert.equal(convertToPennies("£p"), -1);
});
```

3 Clean Amount Text test: this function removes the '£' & 'p' chars from the user input and returns the clean value without these signs if there is any. All tests were successful.

```
QUnit.test("Clean Amount Text test", function(assert) {
  assert.equal(cleanAmountText("£12.34"), "12.34");
  assert.equal(cleanAmountText("123p"), "123");
  assert.equal(cleanAmountText("432"), "432");
  assert.equal(cleanAmountText("213p"), "213");
  assert.equal(cleanAmountText("£16.23p"), "16.23");
  assert.equal(cleanAmountText("£14"), "14");
  assert.equal(cleanAmountText("£54.04"), "54.04");
  assert.equal(cleanAmountText("£23.33333"), "23.33333");
  assert.equal(cleanAmountText("001.41p"), "001.41");
  assert.equal(cleanAmountText("13x"), -1);
  assert.equal(cleanAmountText("13p.02"), -1);
  assert.equal(cleanAmountText("£p"), -1);
});
```

4 Is Valid Input test: this function checks the input value if it meets the input predefined pattern or not. All tests were successful.

```
QUnit.test("Is Valid Input test", function(assert) {
  assert.equal(isValidInput("£12.34"), true);
  assert.equal(isValidInput("123p"), true);
  assert.equal(isValidInput("432"), true);
  assert.equal(isValidInput("213p"), true);
  assert.equal(isValidInput("£16.23p"), true);
  assert.equal(isValidInput("£14"), true);
  assert.equal(isValidInput("£54.04"), true);
  assert.equal(isValidInput("£23.33333"), true);
  assert.equal(isValidInput("001.41p"), true);
  assert.equal(isValidInput("13x"), false);
  assert.equal(isValidInput("13p.02"), false);
  assert.equal(isValidInput("£p"), false);
});
```

To view the test results, go to the following link in your preferred browser:

- On Windows: http://localhost/CoinsAmount/unitTesting.html
- On MAC: http://192.168.64.2/CoinsAmount/unitTesting.html

The following page should show all the test cases results:

User Input Parsing

The user input is being checked using a Regex pattern. The created pattern allows optional '£' sign in the beginning and 'p' letter to the end of the input. In addition, it would accept any numerical values with or without floating point.

The used Regex pattern is: "^(£?[0-9]+(?:.?[0-9]+))(p)?$" and it can be used by declaring the pattern into a variable as follow:

var InputPattern = new RegExp("^(£?[0-9]+(?:.?[0-9]+))(p)?$");

We can use this patter to check any input value by calling the "test" function as follow:

InputPattern.test("£12.5");

The "test" function will return a true/false value according to the pattern declared to do the test.