Summary: Develop detailed documentation that guides developers through the process of integrating Chimoney Payouts into their applications and platforms. You can use a demo project for this

Idea Flow:

Explain the benefits and use cases of Chimoney Payouts.

Provide code examples and step-by-step integration instruction(using your language of choice).

**Chimoney Payouts Integration Guide**

**Benefits of Chimoney Payouts:**

Efficiency: Streamline payout processes for businesses and platforms.

Global Reach: Enable cross-border transactions with ease.

Developer-Friendly: Simple API integration for seamless implementation.

Secure Transactions: Ensure the security of financial transactions.

Versatility: Applicable across various industries and use cases.

**Use Cases:**

E-commerce Platforms: Swiftly pay vendors and affiliates globally.

Gig Economy Apps: Facilitate quick and secure payments to freelancers.

Financial Apps: Enable seamless money transfers between users.

Marketplaces: Efficiently distribute funds to sellers or service providers.

Crowdfunding Platforms: Disburse funds to project backers with ease.

Remittances: Facilitate cross-border money transfers for users.

Subscription Services: Automate recurring payments for subscribers.

Employee Payments: Streamline payroll processes for businesses.

Digital Services: Pay content creators, influencers, or contributors effortlessly.

Non-profit Organisations: Distribute funds to beneficiaries or support initiatives.

Integration Steps (Using javascript and html as an Example):   
Finished product: https://rb.gy/y5awd

Step 1: Obtain API Key

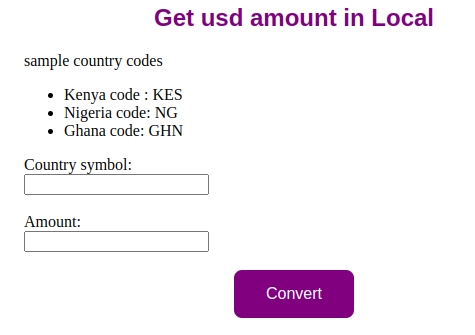
Register on Chimoney and obtain your API key in the developers portal t

Follow this tutorial on how to get the get started with chimoney api and developer portal <link>

For this tutorial we will be using the convert local currency amount to USD endpoint this is the url for that https://api.chimoney.io/v0.2/payouts/mobile-money

Step 2: user interface

We will create a small interface for easier integration with our api key the interface will provide a way for us to pass the two parameters that is the amount and the country code that are required for the endpoint.

  
<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Document</title>

<script src="./apifetch.js"></script>

<script src="./key.js"></script>

<link rel="stylesheet" href="./style.css">

</head>

<body>

<div class="title">

<h2>Get usd amount in Local</h2>

</div>

<div class="country">

sample country codes

<ul>

<li>Kenya code : KES</li>

<li>Nigeria code: NG</li>

<li>Ghana code: GHN</li>

</ul>

</div>

<div class="form">

<label>Country symbol</label>

<input type="text" id="symbol"> <br/> <br/>

<label>Amount </label>

<input type="text" id="amount"><br/><br/>

</div>

<div class="button">

<button onclick="convert()">Convert</button>

</div>

<div id="result"></div>

</body>

</html>

The major parts of this code is this line <button onclick="convert()">Convert</button>

Which on click calls the convert method which is where we have our api function

Step 3: Initialize Chimoney

Let's now dive into the real part we will be using javascript for this and use the fetch in built method  
function convert () { //function triggered on click the

const countrySymbol = document.getElementById('symbol').value; // this are to passed as parameter to the endpoint and the are keyed in by the user

const amountInUSD = document.getElementById('amount').value;

const options = { // fetch method takes to params that is the options and url so lets define the options

method: 'GET', // this is the method used get method retrieves data

headers: { // takes in our api key as application json

accept: 'application-json',

'X-API-KEY': "f53b14db7658d93aeb2f8589838b87e01d730d82b9c73e9878065452902790b7" //for best practices save your api key in the .env file

}

};

const apiUrl = 'https://api.chimoney.io/v0.2/info/usd-amount-in-local?destinationCurrency=USD&amountInUSD=1' //pass the url to the endpoint where data is to bne fetched

fetch(apiUrl, options) //as we said earlier fetch takes two params pass them here, you can also use the axios method.it's also safe to test your codes in the sandbox/postman before coding it

.then(response => response.json())

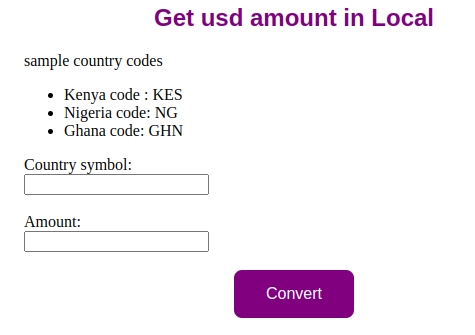
.then(response => { document.getElementById('result').innerText = `Result: ${JSON.stringify(response)}`; ///we stringify the response we get so that it becomes user friendly then we render it under the result div

})

.catch(err => console.error(err));//if the response is not of status 200 a descriptive error will be returned

}

Step 4: Make a conversion

Click on the covert button and the result will be displayed  
  


Step 5: Handle Errors

Implement error handling to manage API responses:

javascript

.catch(err => console.error(err)); //this will make it easier to debug your code

**Demo Project:**

link to GitHub repository containing the demo project showcasing the integration steps and a working example.  
https://github.com/Davidongora/currency\_converter