

Problem 2: Fancy Form

Contents

1. OVERVIEW	3
2. CONFIGURE PROJECT	6

1. OVERVIEW

Link demo:

<https://currency-swap-ten.vercel.app/>

Technology:

- NextJS, TypeScript, TailwindCSS.

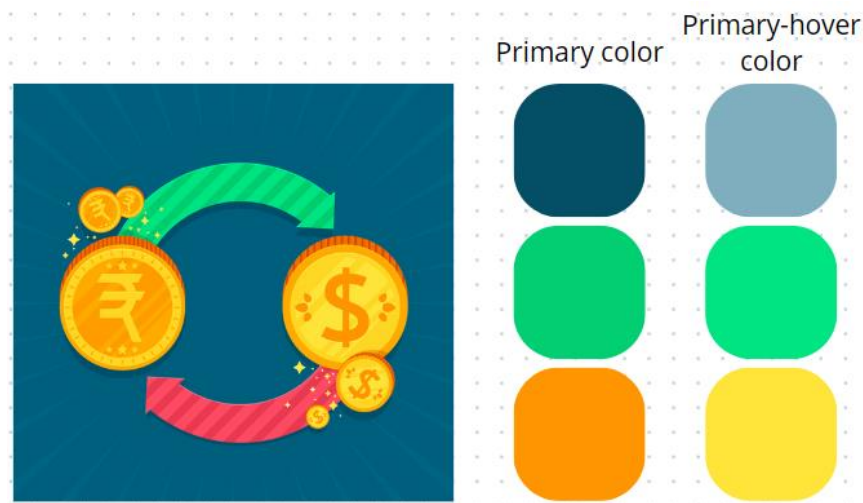
Flow:

Note: I don't have enough time to draw a Sequence diagram. So I will present step by step.

1. User input amount to send and select token to swap.
2. Send 3 variables to the system:
 - amountToSend
 - sendTokenId
 - receiveTokenId.
3. Call API to get token prices base on sendTokenId, receiveTokenId.
4. Calculate swapping.
5. Display result for user.

Template:

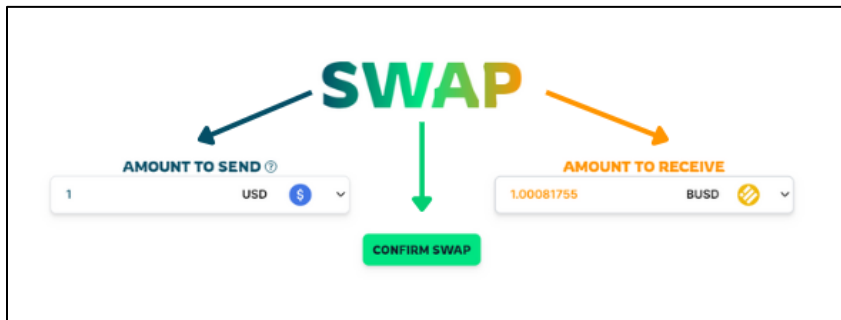
- I searched for ideas “currency swap” and I chose this photo. By standard, a website should only have 3 colors. so I picked out 3 primary colors and 3 primary-hover colors.



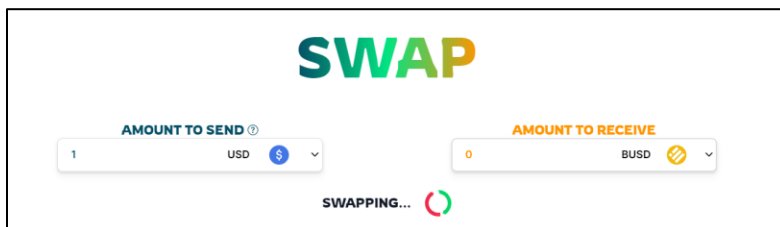
- I searched for fonts for the entire website on google fonts and I choose **Rowdies font**.



Layout ideas:



Loading page at first render and timeout delay for the submit button:



Layout Responsive:

Desktop Monitor, Laptop:

The image shows the SWAP interface on a desktop monitor or laptop. At the top, the word "SWAP" is displayed in a large, bold, green and yellow font. Below it, there are two input fields. The first field, labeled "AMOUNT TO SEND" with a question mark icon, contains the value "1" and a dropdown menu showing "USD" with a blue dollar sign icon. The second field, labeled "AMOUNT TO RECEIVE", contains the value "1.00081755" and a dropdown menu showing "BUSD" with a yellow coin icon. A green button labeled "CONFIRM SWAP" is positioned below the input fields.

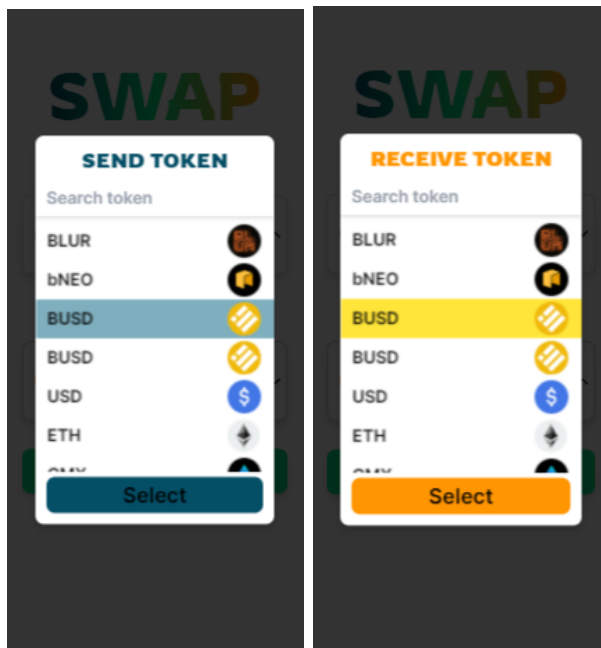
With devices Desktop Monitor, Laptop I setting Hover:

The image shows the SWAP interface on a desktop monitor or laptop with a hover effect. The "AMOUNT TO SEND" field now shows "0.00" and the "AMOUNT TO RECEIVE" field shows "0". The "AMOUNT TO RECEIVE" dropdown menu is open, showing a search bar labeled "Search token" and a list of tokens: "BLUR" (highlighted in yellow), "bNEO", "BUSD", and "BUSD". A green button labeled "CONFIRM SWAP" is positioned below the input fields.

Mobile, Tablet:

The image shows the SWAP interface on a mobile device or tablet. The "SWAP" logo is at the top. Below it, the "AMOUNT TO SEND" field shows "0.00" and the "AMOUNT TO RECEIVE" field shows "0". Both fields have a dropdown menu showing "USD" with a blue dollar sign icon. A green button labeled "CONFIRM SWAP" is positioned below the input fields.

With devices Mobile don't have action hover so I design Select button.

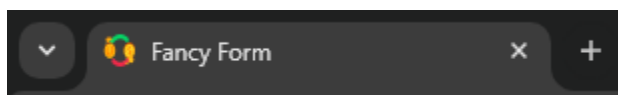


2. CONFIGURE PROJECT

Configuration in Tailwind:

```
colors: {
  "fancy-bg-blue-dark": "#044E66",
  "fancy-bg-blue-hover": "#7FAFBE",
  "fancy-bg-blue": "#005F7D",
  "fancy-green": "#00E482",
  "fancy-green-dark": "#02CE72",
  "fancy-yellow-hover": "#FFE53A",
  "fancy-yellow-dark": "#FF9500",
  "fancy-pink": "#FE4C62",
  "fancy-pink-dark": "#EA3852",
},
fontFamily: {
  Rowdies: ["Rowdies", "sans-serif"],
},
```

Configuration favicon:



Mock data and configuration data type.

I split data to listToken and Prices because when dropdown listToken to select, we don't need to call API to get price.

- listToken data is a list of tokens that users can exchange

```
export const listToken = [
  {
    token_id: "token1",
    currency: "BLUR",
  },
  {
    token_id: "token2",
    currency: "bNEO",
  },
  {
    token_id: "token3",
    currency: "BUSD",
  },
]
```

- Prices data is the price of the token when calling the API

```
export const Prices = [
  {
    token_id: "token1",
    currency: "BLUR",
    date: "2023-08-29T07:10:40.000Z",
    price: 0.20811525423728813,
  },
  {
    token_id: "token2",
    currency: "bNEO",
    date: "2023-08-29T07:10:50.000Z",
    price: 7.1282679,
  },
  {
    token_id: "token3",
    currency: "BUSD",
    date: "2023-08-29T07:10:40.000Z",
    price: 0.999183113,
  },
]
```

- Type of data

```
export interface IListToken {
  token_id: string;
  currency: string;
}

export interface ITokenInfo extends IListToken {
  date: string;
  price: number;
}
```

- Fake call API (Path: [src\app\utils\Swap.ts](#))

```
const sendTokenInfo = Prices.find(
  (token: ITokenInfo) => token.token_id === sendTokenId
);
const receiveTokenInfo = Prices.find(
  (token: ITokenInfo) => token.token_id === receiveTokenId
);
```