Simple logic
Please create solutions from these cases:
1. Check the string is palindrome or not

```
Input:
isPallindrome(‘abcba’)

Result:
true
```

2. Find prime number by range

```
Input:
findPrimeByRange(11, 40)

Result:
[11, 13, 17, 19, 23, 29, 31, 37]
```

3. Grouping array group into separate sub array group

```
Input:
const arr = [‘a’, ‘a’, ‘a’, ‘b’, ‘c’, ‘c’, ‘b’, ‘b’, ‘b’, ‘d’, ‘d’, ‘e’, ‘e’, ‘e’]
group(arr)

Result:
[ [‘a’, ‘a’, ‘a’], [‘b’], [‘c’, ‘c’], [‘b’, ‘b’, ‘b’], [‘d’, ‘d’], [‘e’, ‘e’, ‘e’] ]
```

4. Count same element in an array with format

```
Input:
const arr = [‘a’, ‘a’, ‘a’, ‘b’, ‘c’, ‘c’, ‘b’, ‘b’, ‘b’, ‘d’, ‘d’, ‘e’, ‘e’, ‘e’]

Result:
[ {3, ‘a’}, {1, ‘b’}, {2, ‘c’}, {3, ‘b’}, {2, ‘d’}, {3, ‘e’} ]
```

Simple web-apps

Backend

- Create API Using Golang Language
- API Register body has username, email, password (Hashed)
- API Login using System Auth token, for the method its up to you, example (bearer token, basic auth, etc)
- API CreateFinancing body has name, count, sub → dummy data

```
{
  "name": "Invoice Financing",
    "count": 35,
    "sub":null,
  },
  {
    "name": "OSF Financing",
    "count": 25,
    "sub":null,
  },
  {
    "name": "SBN",
    "count": 10,
    "sub":null,
  },
  {
    "name": "Reksadana",
    "count": 20,
    "sub":null,
  },

 {

    "name": "Conventional Invoice",
    "count": 20,
    "sub": "invoice"
  },
  {
    "name": "Productive Invoice",
    "count": 15,
    "sub": "invoice"
  }

  {

    "name": "Conventional OSF",
    "count": 15,
    "sub": "osf"
  },
  {
    "name": "Productive OSF",
    "count": 10,
    "sub": "osf"
  }
```

- API CreateConventionalOsf body has name, amount, tenor, grade, rate
- API CreateConventionalInvoice body has name, amount, tenor, grade, rate
- API CreateProductiveInvoice body has name, amount, grade, rate
- example

```
   {
    "name": "PT YJK",
    "amount": 1000000,
    "tenor": 120,
    "grade": "B",
    "rate": 16
  },


  {
    "name": "PT KKY",
    "amount": 4000000,
    "tenor": 120,
    "grade": "B+",
    "rate": 14
  }
```

- API Create Reksadana body has name, amount, return
- example

```
  {
    "name": "INB",
    "amount": 20000000,
    "return": -1
  },
  {
    "name": "PT TELMOM",
    "amount": 10000000,
    "return": 1
  },
```

- API Create sbn body has name, amount, tenor, rate, type
- example

```
   {
      "name": "SBR XXX",
      "amount": 1000000,
      "tenor": 120,
      "rate": 7,
      "type": "SBR"
    },
    {
      "name": "SBR YYY",
      "amount": 2000000,
      "tenor": 120,
      "rate": 8,
      "type": "SBR"
    },
```

- All create API must have API get

- For Database its up to you

- All this system installed on Container Docker

Frontend

Please create a web application using component-based JS Framework, (we recommend Vue.js) using provided JSON. Also please create app as close as possible with provided illustration.

App Requirement:

- Clickable menu if the item still has sub-directory in it
- Create login menu and register menu (design up to you)
- Design homepage like the ingredients page, page1, page2
- All data get from API you already build on Backend Test
- If the items don’t have sub-directory anymore, create filter-select with requirements:
  - ➢ Loan page (Invoice & OSF): create 1 select-filter, that filter based on object’s grade (A, B+, B)
  - ➢ SBN: create 1 select-filter that filter based on object’s type (SBR, ST)
  - ➢ Reksadana: create 1 select-filter, based on object’s rate, positive (>= 0), negative (< 0)
  - ➢ Remember every filter should have “filter-none” or you can call it “All” options, which means filter is not applied
- If the items don’t have sub-directory anymore, create search that will filter items basedon object’s name
- All filters must be working together, which means if user use select-filter and search, the data will be filtered by ‘select-filter’ AND ‘search’
- Infinite scroll (show 5 items per scroll), with initial showing 5 data (do not forget to add some animation when page still loading new items)

Code Requirement:

- Please do consistent coding convention, like following Airbnb ESLint config for example (https://www.npmjs.com/package/eslint-config-airbnb)
- Good folder and file management (including naming)
- You can use bootstrap for grid management, but we encourage you create your custom CSS rules for the other things
- Good componentization will be assessed
- You may use 3 rd party plugin, but remember, good hand-crafted JavaScript code will always have special place for assessment

Submission:

Push your source code in git repository and then send us the link. Push into gitlab (mandatory).
Please add our gitlab account (reza.dompetkilat) into `developer` in your repository

- Try documenting your code, don’t need to be that fancy, just simple and direct documentation is enough
- Please give us the README file. How to run the program. Good README file will be assessed
- Commit history to git will be assessed. Please do commit your code part by part

Hint:

- Forget that old one-page coding, see every section as a component to be made
- Do not hesitate to ask, you can reach me on reza.basuki@dompetkilat.id
- For icons you can refer to this link https://fontawesome.com/cheatsheet