C1 - MVVM frameworks

C-DEV-121

Post-it

notes management application

2.1

Post-it

delivery method: postit on Github
language: Vue.js

- The totality of your source files, except all useless files (binary, temp files, obj files,...), must be included in your delivery.

Refer to the general documentation of how to write a README.md in your courses to improve the quality of your project's delivery.

Let's create a front-end application using Vue.js to discover this technology.

Your application will allow you to manage post-it notes.

Goals

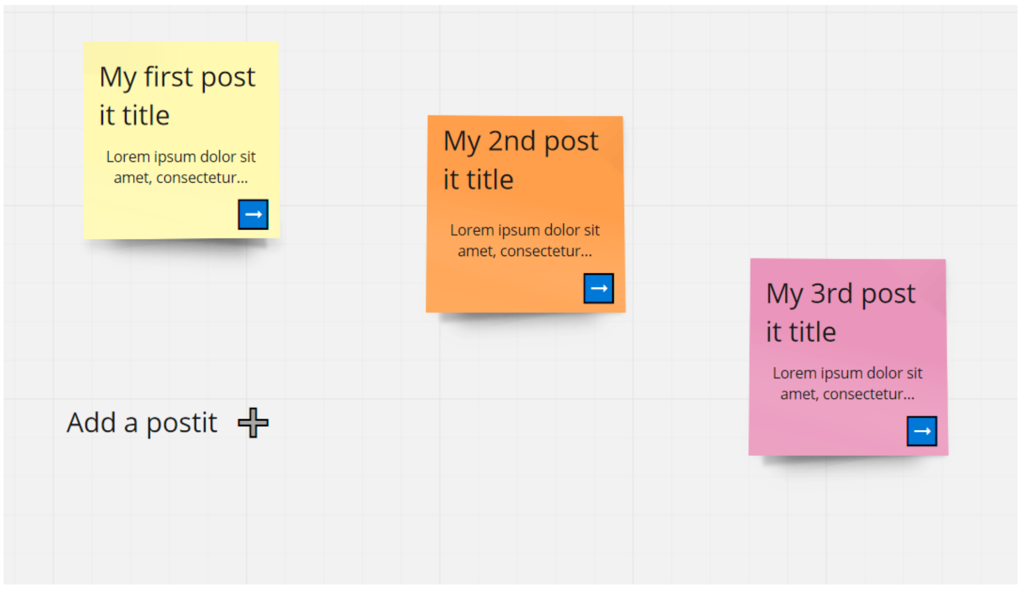
This project will allow you to acquire the following notions:

- Creating Vue.js component
  - Data attribute of a component
  - Methods attribute of a component
  - Computed attribute of a component
  - Created attribute of a component
  - Link between data and views
  - Component life cycle
- Implementation of a model allowing interaction with data
  - Props
  - Issue and retrieve events
  - Handling user input
  - Use of LocalStorage

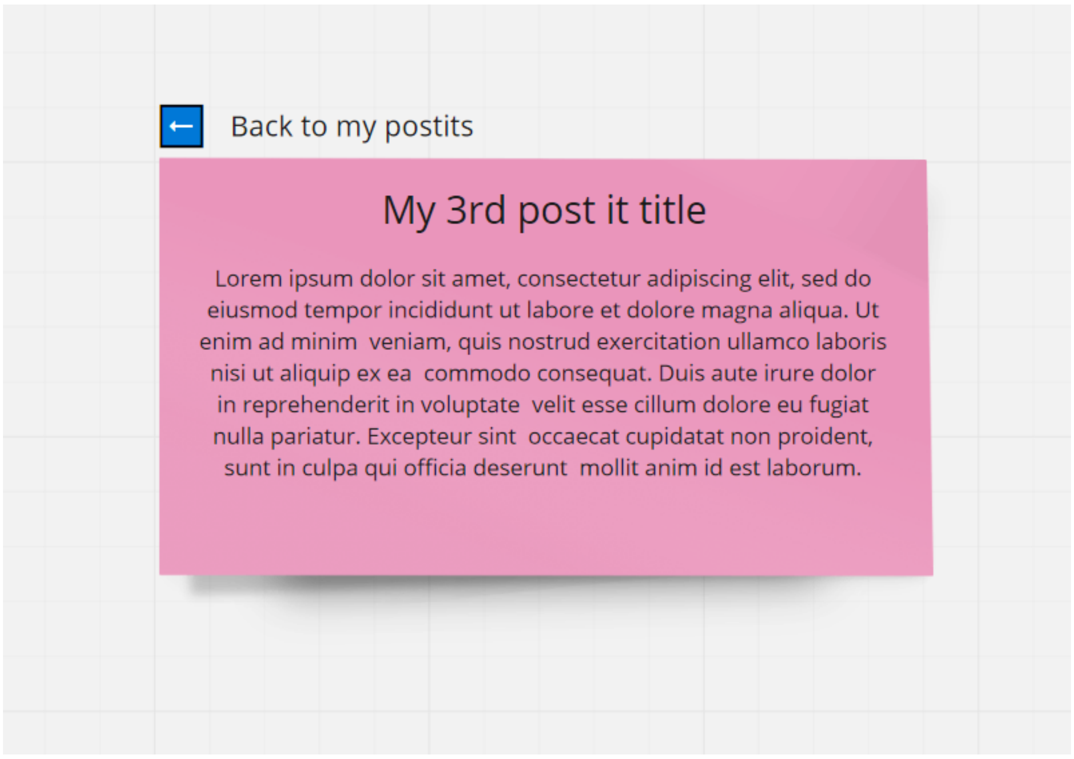
We advise you to use a linting tool

Step A1. Views

- Create a view that will display the items on your post-it notes list.


My first post it title
Lorem ipsum dolor sit amet, consectetur...
My 2nd post it title
Lorem ipsum dolor sit amet, consectetur...
My 3rd post it title
Lorem ipsum dolor sit amet, consectetur...
Add a postit

- Create a view that will display information for a selected post-it note.



My 3rd post it title

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Only HTML template, no real data at this moment

STEP A2. ROUTER

- You will create two routes:
    - "/" which will display your "Notes" component allowing you to display all your post-it notes.
    - "/note" which will take the id of the post-it indicated and display the various information of it in the "Note" component.

STEP A3. DATA

- You will use the localStorage to store all the data.
- For each component, store and update the local data of this component

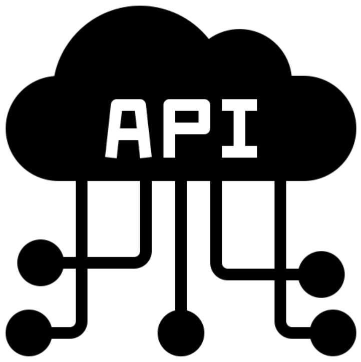
think local first then global !

Step A4. Wrapping-up

- You need to create communication system between your components
- You can now link the data to the views, you should have a working post-it application.

you have to use props

You decide to edit your Post-it app in order to upgrade it by storing the data on the internet.



You will be able to store and retreive Post-it's data online using an API, rather than the localStorage.

we advise you to use a linting tool

You will use ES6 Fetch to make API requests

Step B1. API protocol

First, get familliar with API by reading it and make requests with `postman` or `curl`.

API's endpoint is `http://5.135.119.239:3090/` and routes' details are provided in another file.

Step B2. Vue Router

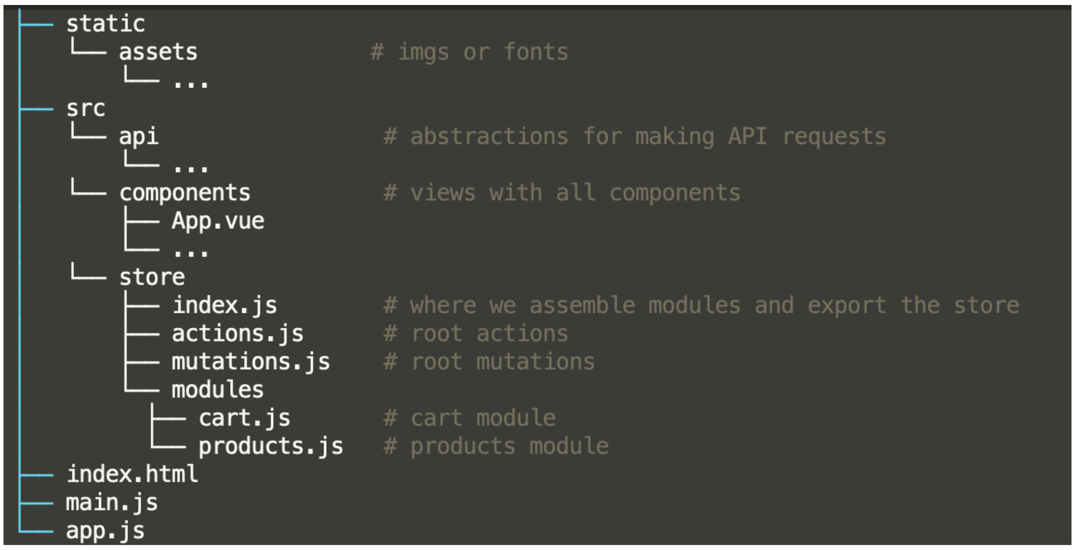
Edit your code to use official Vue.js's router

Step B3. Vuex State Management

Add to your code a state management system.

You will use Vue.js's official package Vuex

You should follow vue.js's project structure



```
├── static
│   └── assets                # imgs or fonts
│       └── ...
├── src
│   └── api                    # abstractions for making API requests
│       └── ...
│   └── components             # views with all components
│       ├── App.vue
│       └── ...
│   └── store
│       ├── index.js           # where we assemble modules and export the store
│       ├── actions.js         # root actions
│       ├── mutations.js       # root mutations
│       └── modules
│           ├── cart.js        # cart module
│           └── products.js    # products module
├── index.html
├── main.js
└── app.js
```

Step B4. Using the API

Edit your component to fetch and send data to/from the API using Fetch.

Your data is saved by the API in a database.

Use Vuex store to store data locally

Bonus

- use a CSS framework
- responsiveness
- add relevant API functionality
- add unit tests
- delicious design
- ...