



サンプルアプリ GoogleBooksAPI



サンプルの紹介

読書管理システム(簡易)

BookIndex

検索する



Webデザインの現場で使えるVue.jsの教科書

読んだ日: 2020-10-10

感想: おもしろかった。あああ




動かして学ぶ!Vue.js開発入門

読んだ日: 2020-10-11

感想: ためになった



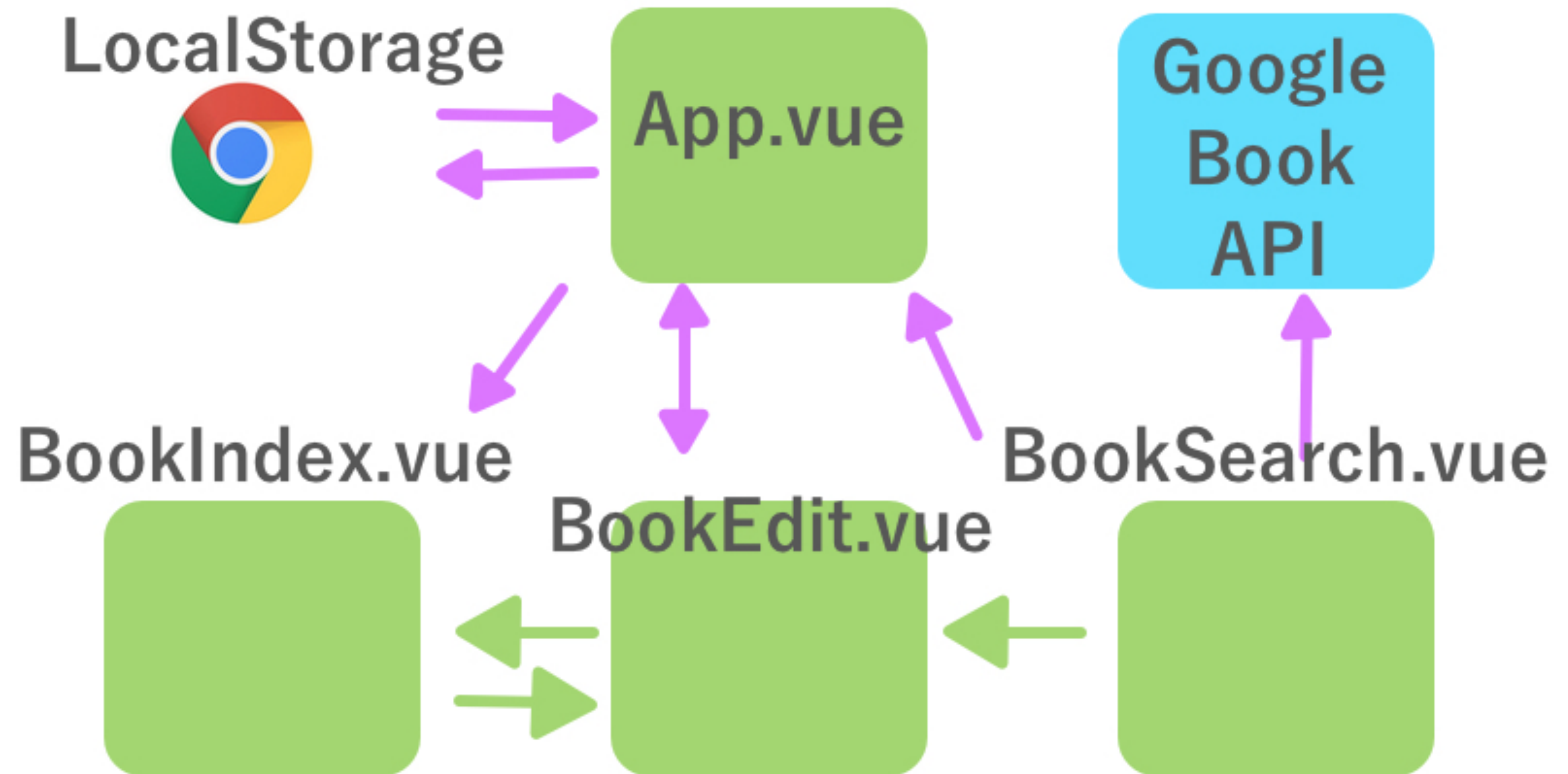
使っている技術



Vue.js2.6.11 / Vuetify2.3.10
VueCLI(SFC) / VueRouter(SPA)

GoogleBooksAPI /
WebStorage(LocalStorage)

簡単な構成図





Google Books API

Google Books API



メリット

登録なしでも使える (1000件/日)

デメリット

検索が少し弱い(表示されない本も)

価格が表示されない・出版日が正確ではない
(AmazonAPIは条件厳しい
(30日以内に売上必要))

Google Books API



ベースのURL

[https://www.googleapis.com/books/v1/volumes?
q=検索語句](https://www.googleapis.com/books/v1/volumes?q=検索語句)


intitle: 本のタイトル

maxResults:40 検索表示数(10-40)

<https://developers.google.com/books>

Guides->Using the API-> Query parameter
reference

クエリーストリング



```
const baseUrl = 'https://www.googleapis.com/  
books/v1/volumes?'
```

```
const params = {  
  q: `intitle:${keyword}`,  
  maxResults:40  
}
```

```
queryParams = new URLSearchParams(params)  
fetch(baseUrl + queryParams)
```



WebStorage (LocalStorage)

Cookie & WebStorage

	サイズ	サーバー通信	有効期限	範囲
クッキー	4KB	毎回	指定期限まで	
Local Storage	1オリジンあたり 5MB	通信しない (必要時のみ)	なし	オリジン単位
Session Storage	1オリジンあたり 5MB	通信しない (必要時のみ)	ウィンドウを 閉じるまで	セッション単位

今回はLocalStorageで(DBの代わり)

LocalStorage



// 取得

`localStorage.getItem(key)`

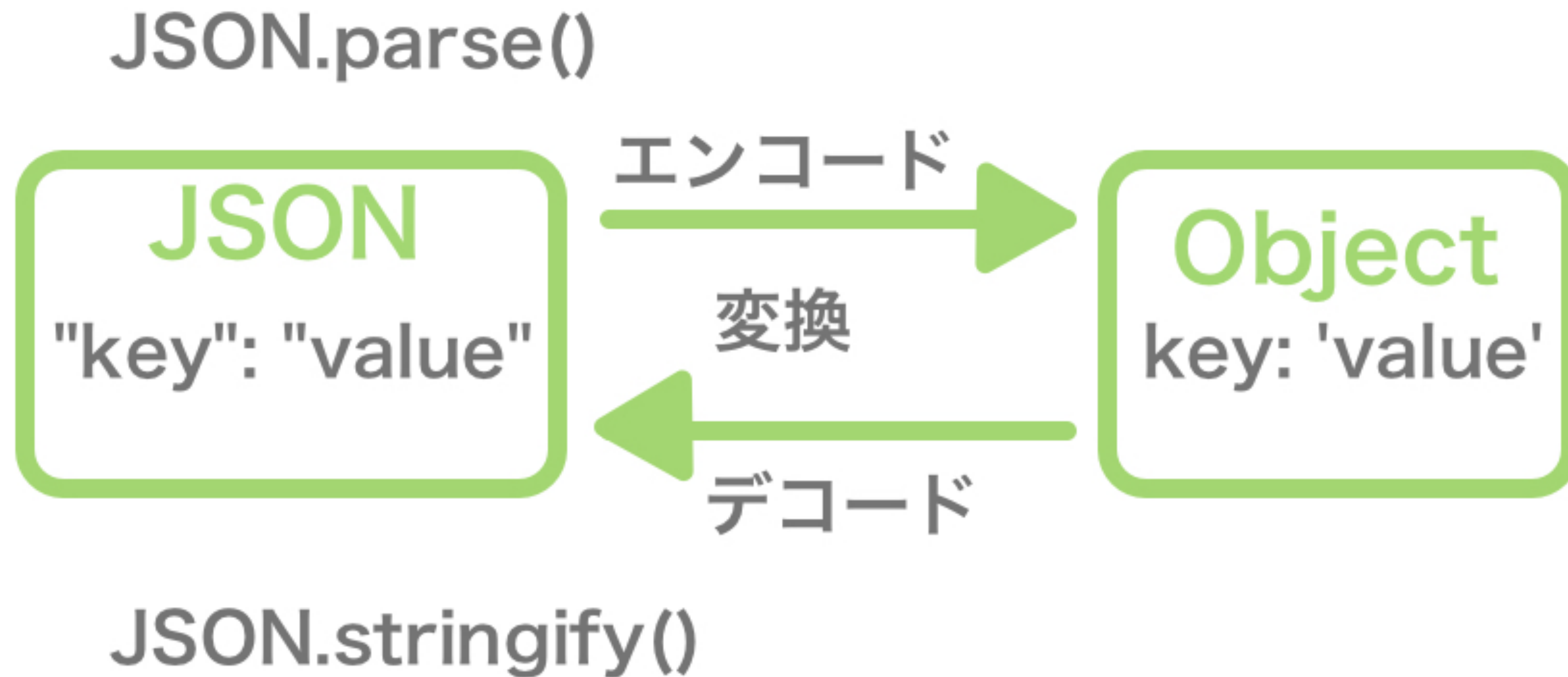
// 保存

`localStorage.setItem(key)`

// 削除

`localStorage.removeItem(key)`

JSONエンコード/デコード



LocalStorage と JSON

```
// 取得 JSON -> Object  
JSON.parse(localStorage.getItem(key))
```

```
// 保存 Object -> JSON  
const parsed = JSON.stringify(Object)  
localStorage.setItem(key, parsed)
```

Vue.jsガイド->クックブック->クライアントサイド
ストレージ



Vuetify

インストール


Vuetify インストール



`npm install —save-dev vuetify` //インストールのみ
`vue add vuetify` //ファイル書き換え含む


プロジェクトに追加できるようにファイルに記述

`main.js`, `plugin/vuetify.js`, `App.vue`
`components/HelloWorld.vue` など



ファイル・フォルダ 構成

ファイル・フォルダ構成



src

App.vue

main.js

router/index.js

global/

Header.vue

Footer.vue

pages/

BookIndex.vue

BookSearch.vue

BookEdit.vue



コンポーネント作成

BookSearch.vue



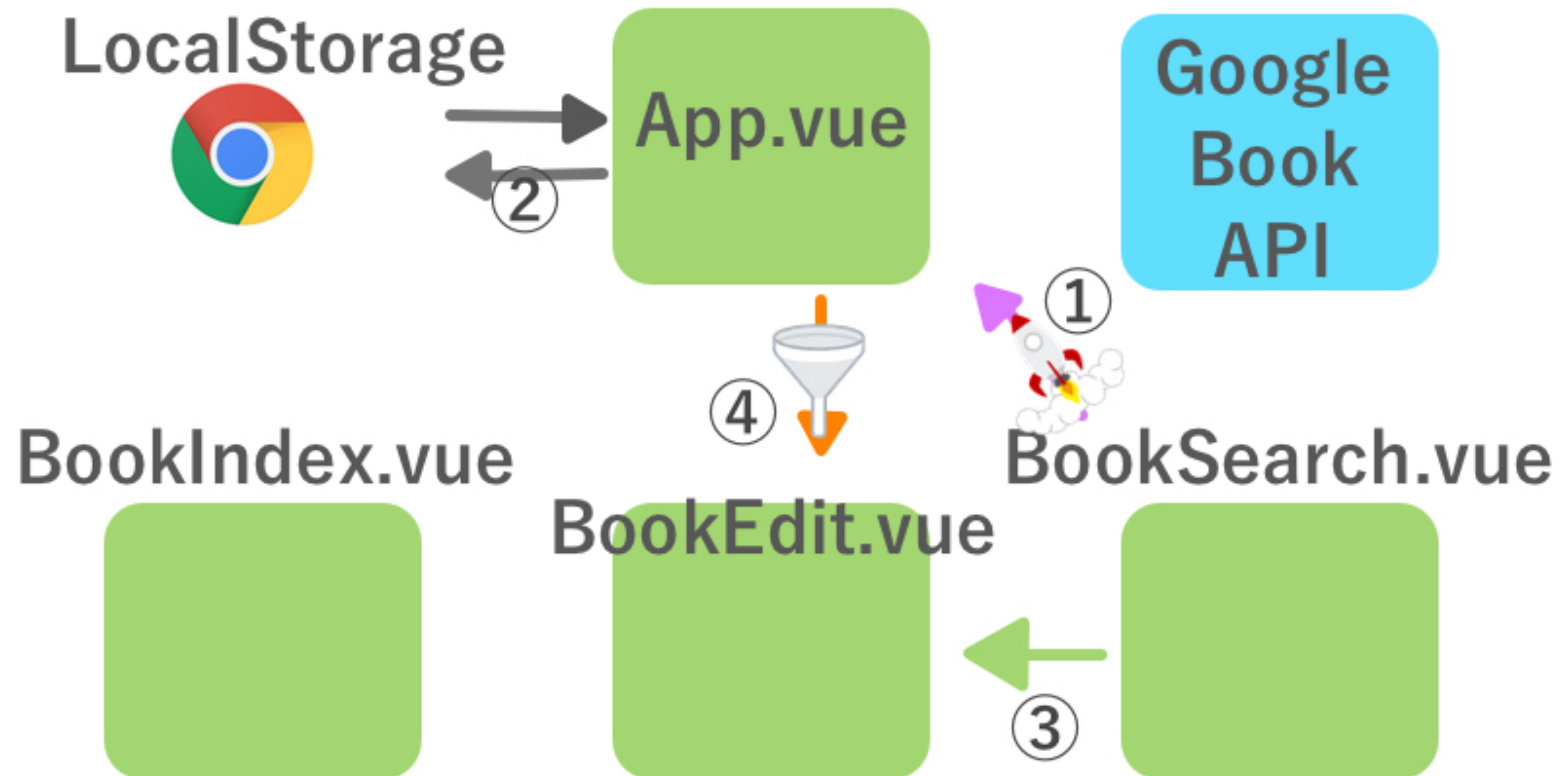
`<v-btn to="/">` //toでrouter-linkと同じ動き

`<v-for="book, index">`

`<v-btn @click="addBookList(index)>`

定義したindexは設定しないとエラーがでるので
@clickでメソッドを設定しておく

LSに追加、 BookEditに移動

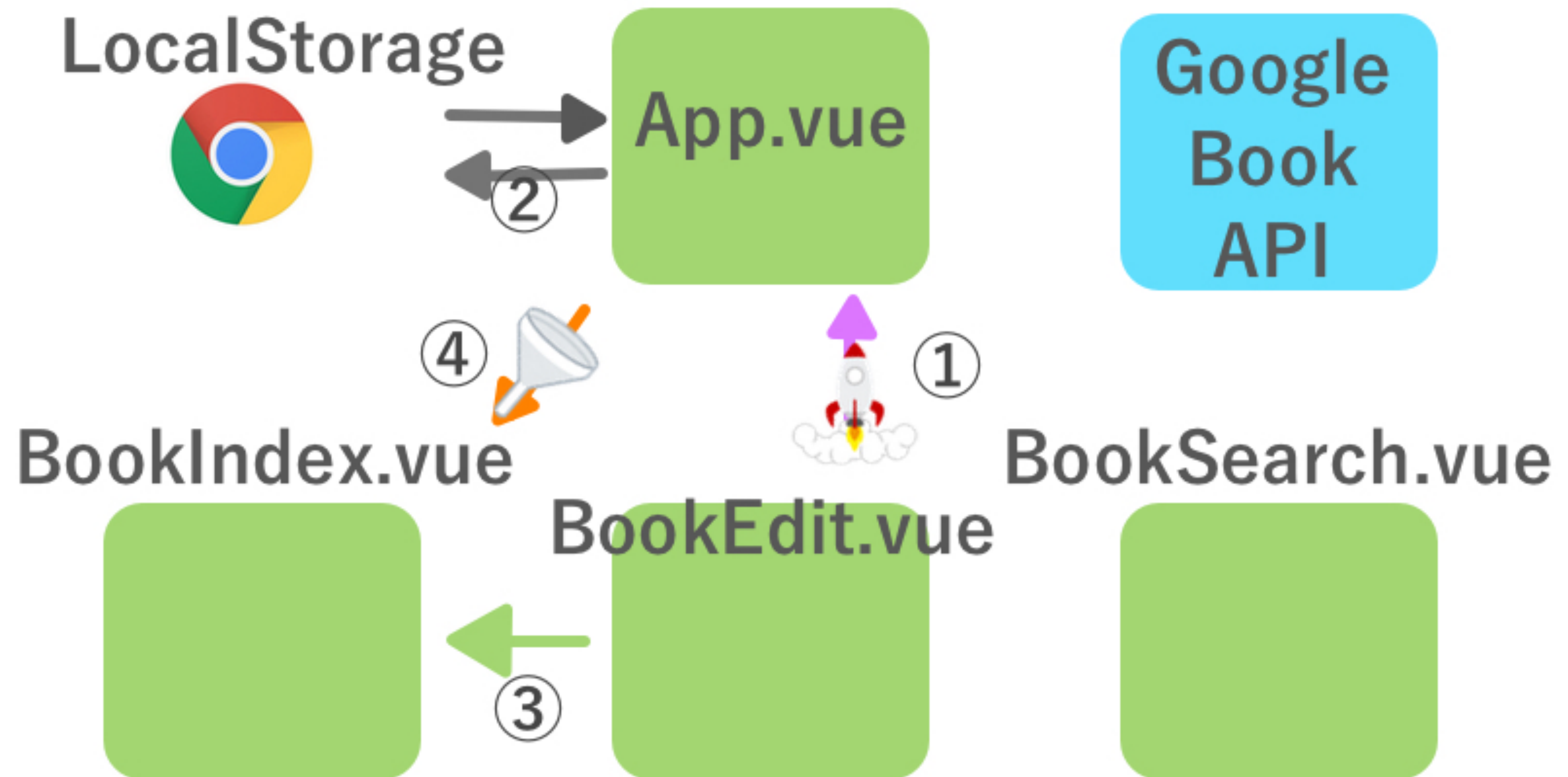


コンポーネントの読み込みは非同期

```
BeforeRouteEnter(to, from, next){  
  next( vm =>{  
    vm.$nextTick(()=>{  
      vm.book = vm.books[vm.$route.params.id]  
    })  
  })  
}}
```

next()の中にさらに\$nextTickを使うと
DOM更新を待ってから処理する
next(vm => {}) で vmを thisのように使える

BookEditで保存後Indexに移動



配列の置換はspliceで

オブジェクトごと置き換わるので
あらかじめオブジェクトをつくる

```
updateBookInfo(e){ // eで子どもからデータ取得
  const updateInfo = {
    id: e.id,
    readDate: e.readDate,
    memo: e.memo,
    title: this.books[e.id].title //imageなども必要です
  }
```

```
this.books.splice(e.id, 1, updateInfo)
}
```


BookEdit.vue dateの改善



```
vm.$nextTick(() =>{  
  if( vm.book.readDate){  
    vm.date = vm.book.readDate  
  } else {  
    vm.date = new Date().toISOString().substr(0,  
    10)  
  }  
})
```

propsの補足



子側 propsのデータをdataで受けると
初回のみ反映される

リアルタイムに反映させたい場合は
computedで受ける必要がある

LocalStorage削除



```
localStorage.setItem(STORAGE_KEY, '')  
localStorage.removeItem(STORAGE_KEY)
```

サンプルアプリの補足



改善案

- ・重複しないように -> Lodashの_searchなど
- ・削除機能 -> idがずれるので インクリメントidを別で作成しておく
- ・初回登録 保存->編集 を 編集->保存 に
- ・ソート機能->ソート番号を用意
- ・レイアウト調整