

Basic Vue.JS



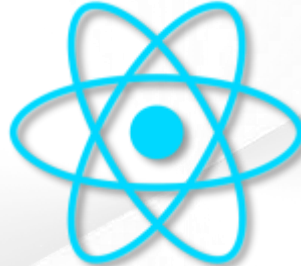


Frontend framework



Angular

- Invented by [Google](#)
- Typescript framework
- As fast as [React](#) fast
- Smart Cli
- Production ready



React

- Invented by [Facebook](#)
- Just js library
- Component-Based
- Fastest famous framework



Vue

- Invented by [Evan You](#)
- JavaScript framework
- Biggest community
- High performance
- Easy to learn and write

ทำไมจึงควรใช้ Vue.js

- ▶ Document อ่านง่าย
- ▶ Component System ใช้งานง่าย แบ่งแยกการทำงานได้ชัดเจน
- ▶ ระบบ Template ของ Vue ใช้งานได้สะดวก
- ▶ สนับสนุนการใช้งานกับ JSX และ HyperScript

```

1 <div id="app">
2   <p>{{ message }}</p>
3   <button v-on:click="reverseMessage">Reverse Message</button>
4 </div>
5

```

- ▶ สารพัดเครื่องมือ เช่น [vue-cli](#) [vue-router](#) [vuex](#) [vue-devtools](#) [vue-loader](#) [vue-resource](#) และ อื่นๆ ที่ช่วยอำนวยความสะดวกให้ทำงานง่ายขึ้น

ข้อเสียของการใช้ Vue.JS

- ▶ บอร์ดกระทู้คำถาม-คำตอบที่น้อย (Stack Overflow)
- ▶ ชุมชนนักพัฒนาที่ยังไม่ใหญ่มากนักเมื่อเทียบกับ React และ Angular

Request Program and IDE

LTS

Recommended For Most Users

Current

Latest Features



Windows Installer

node-v16.17.0.msi



macOS Installer

node-v16.17.0.pkg



Source Code

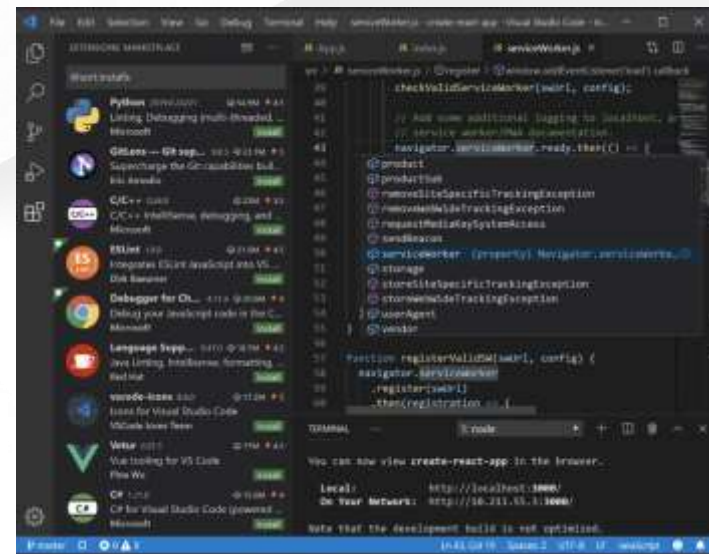
node-v16.17.0.tar.gz

Windows Installer (.msi)	32-bit	64-bit
Windows Binary (.zip)	32-bit	64-bit
macOS Installer (.pkg)	64-bit / ARM64	
macOS Binary (.tar.gz)	64-bit	ARM64
Linux Binaries (x64)	64-bit	
Linux Binaries (ARM)	ARMv7	ARMv8
Source Code	node-v16.17.0.tar.gz	

<https://nodejs.org/en/download/>

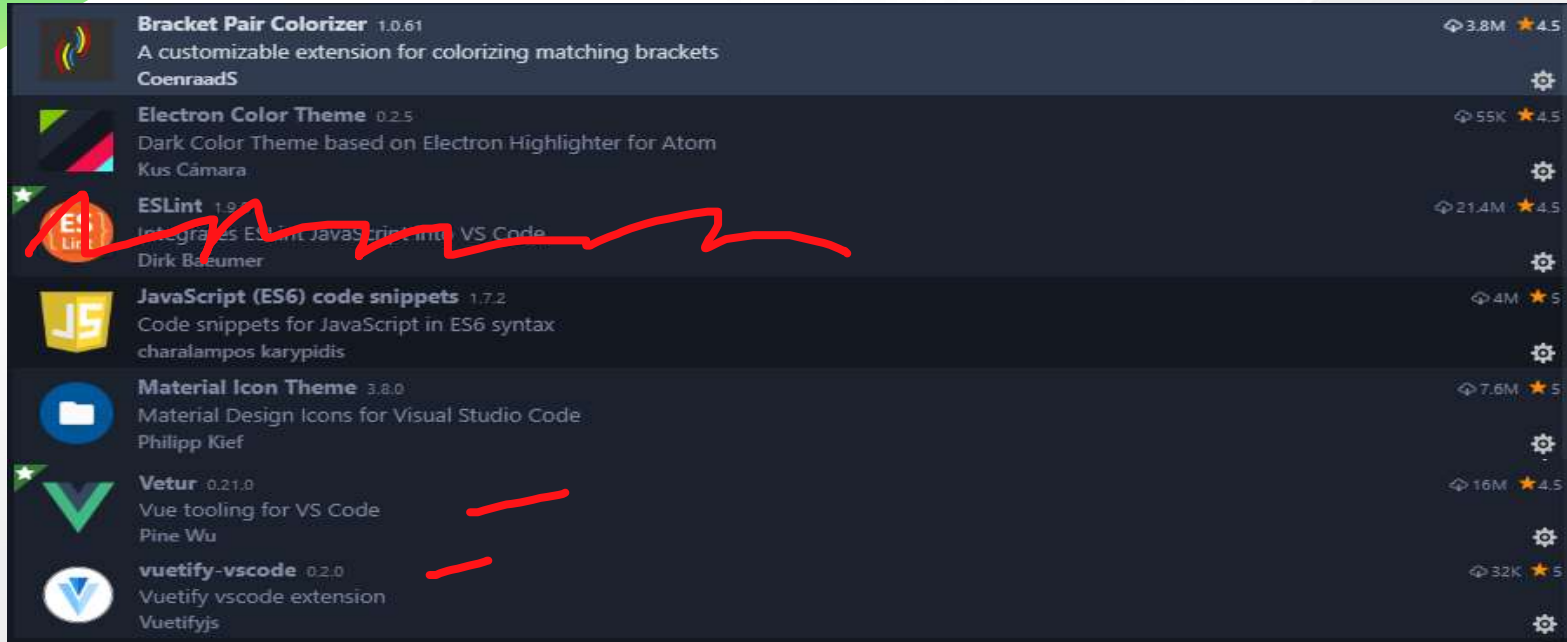
****แนะนำให้ดาวน์โหลด LTS (Long Term Support)**

วิธีคือว่า install สำเร็จ หรือไม่ โดยพิมพ์คำสั่ง `node -v`
ทาง command line



<https://code.visualstudio.com/>

Extension vscode



The screenshot shows the VS Code Extensions view with a list of installed and recommended extensions. A red squiggly line is drawn across the ESLint extension row, and two red horizontal lines are drawn under the Vetur and vuetify-vscode extensions.

Extension Name	Version	Description	Author	Downloads	Rating
Bracket Pair Colorizer	1.0.61	A customizable extension for colorizing matching brackets	CoenraadS	3.8M	4.5
Electron Color Theme	0.2.5	Dark Color Theme based on Electron Highlighter for Atom	Kus Cãmara	55K	4.5
ESLint	1.9.0	Integrates ESLint JavaScript lint into VS Code	Dirk Bäumer	21.4M	4.5
JavaScript (ES6) code snippets	1.7.2	Code snippets for JavaScript in ES6 syntax	charalampos karypidis	4M	5
Material Icon Theme	3.8.0	Material Design Icons for Visual Studio Code	Philipp Kief	7.6M	5
Vetur	0.21.0	Vue tooling for VS Code	Pine Wu	16M	4.5
vuetify-vscode	0.2.0	Vuetify vscode extension	Vuetifyjs	32K	5

Reference

- ▶ <https://www.babelcoder.com/blog/posts/vue2-introduction-to-vue2>
- ▶ <https://www.babelcoder.com/blog/posts/react-vs-vue2-angular2-1>
- ▶ <https://vuejs.org/v2/guide/instance.html>



เริ่มต้นสร้าง Project !

เรียกใช้คำสั่งตามลำดับ

คำสั่ง

`npm i @vue/cli -g`

`npm i @vue/cli-init -g`

~~`npm i yarn -g`~~ ไม่ลงก็ได้

vue create ชื่อโปรเจกต์ เช่น vue create workshop

- เลือก Manually select features
- เลือก Babel, PWA, Router, Vuex, Linter/Formatter
- Use history mode for router? ตอบ Y
- เลือก ESLint + Standard config
- เลือก Lint on save
- เลือก In dedicated config files
- Save this as a preset for future projects? (y/N) ตอบ N

ใช้คำสั่ง (cd ชื่อโปรเจค) เพื่อเข้าไปยัง directory ของโปรเจค



ติดตั้ง Vuetify

คำสั่ง `vue add vuetify`

เลือก default

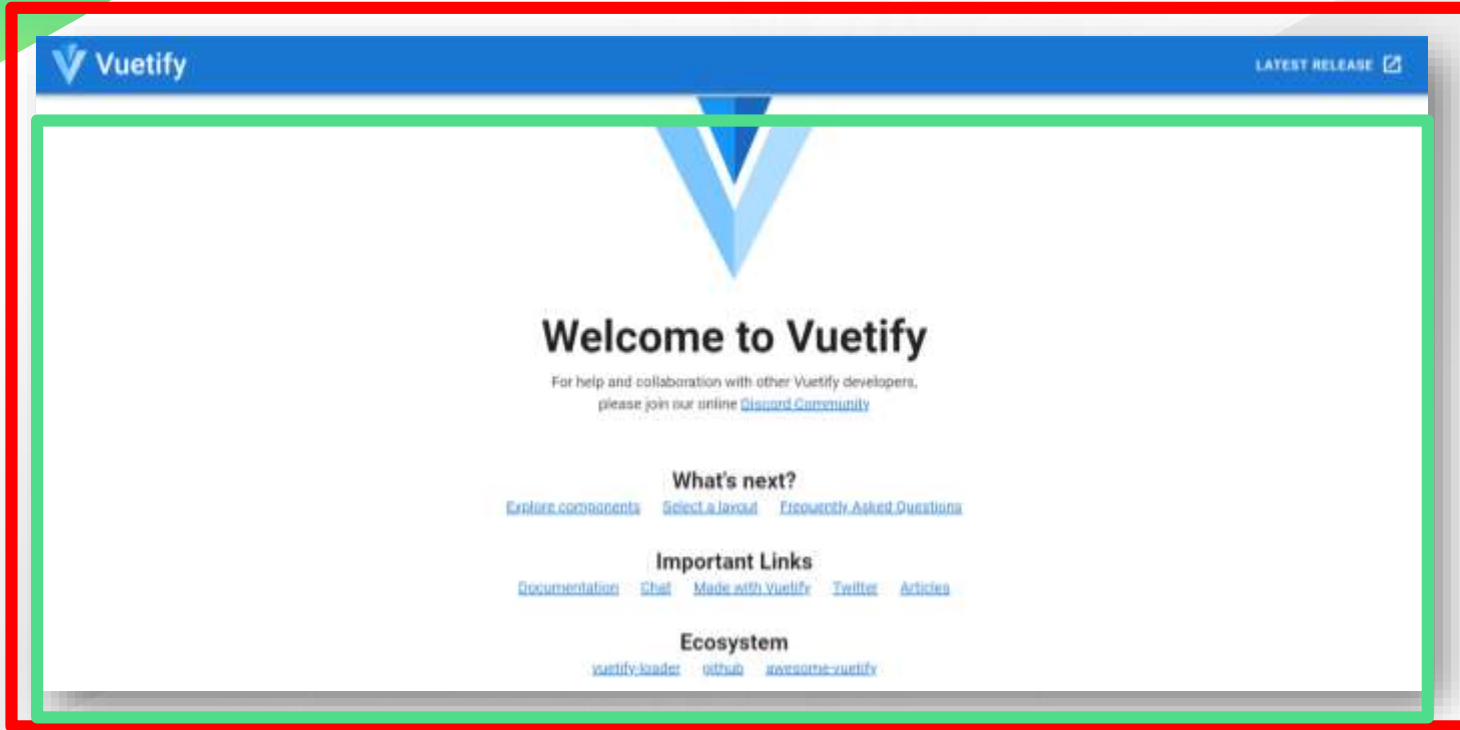
คำสั่งเปิด vscode จาก powershell พิมพ์ code .

คำสั่งรัน `yarn serve`

`npm run serve`

ศึกษา vuetify เพิ่มเติมจาก <https://vuetifyjs.com/en/getting-started/quick-start>

App.vue



components/Helloworld.vue

ไฟล์ .vue ประกอบด้วย

`<template></template>`

tag HTML + สารพัด syntax ของ Vue

เป็นส่วนเอาไว้เพื่อสร้าง UI (ส่วนแสดงหน้าเว็บ)

`<script></script>`

จัดการการทำงานต่างๆ เช่น data component function

`<style></style>`

ใส่ CSS เพื่อตกแต่ง จัดความสวยงามให้กับ components ต่างๆ

Component : ส่วนประกอบ

3.เรียกใช้

1.นำเข้า

2.ประกาศ

```
<template>
  <hello-world />
</template>

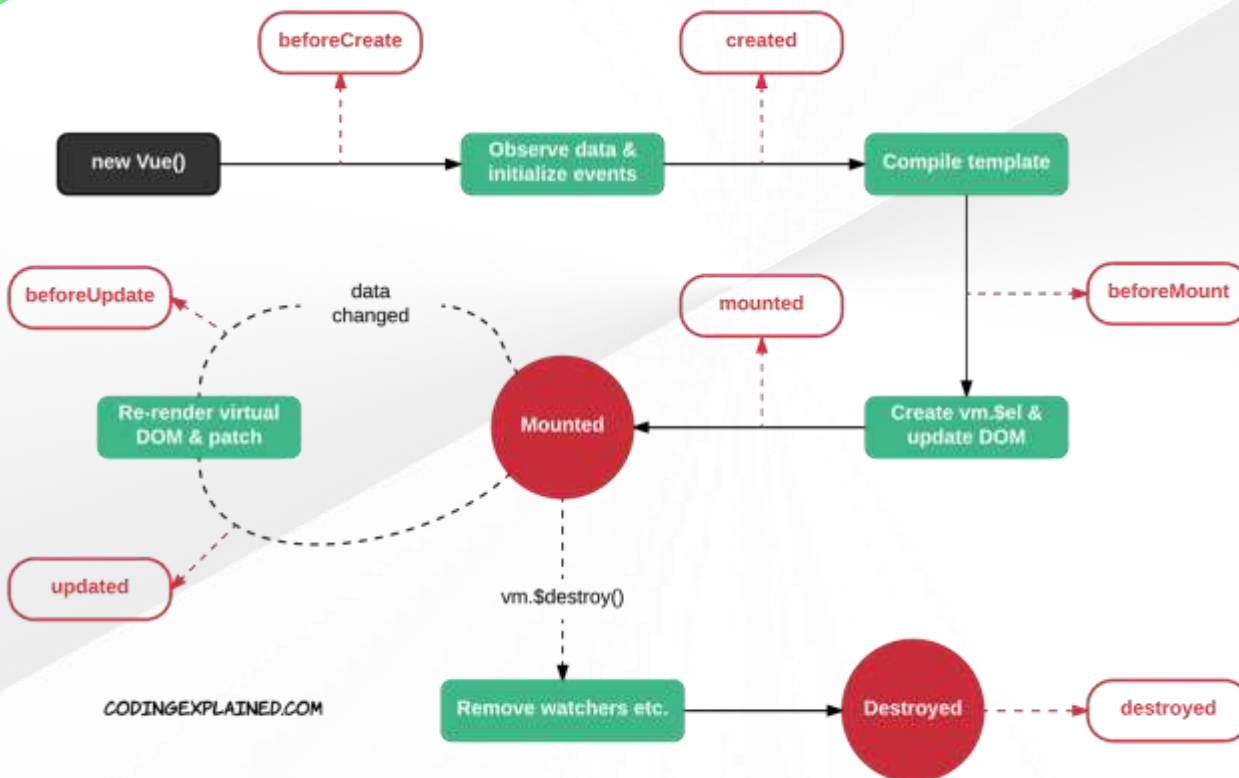
<script>
import HelloWorld from '../components/HelloWorld'

export default {
  name: 'HomeView',

  components: {
    HelloWorld
  }
}
</script>
```

Views/HomeView.vue

Vue.js Lifecycle



Route เป็นการเชื่อมโยงไปยังหน้าต่างๆ

- เพิ่ม path ได้ที่ไฟล์ *index.js*

ใน *folder router*

```
const routes = [
  {
    path: '/',
    name: 'login',
    component: Login
  },
  {
    path: '/home',
    name: 'home',
    component: Home
  },
  {
    path: '/about',
    name: 'about',
    // route-level code-splitting
    // this generates a separate chunk (about.[hash].js) for this route
    // which is lazy-loaded when the route is visited.
    component: () => import(/* webpackChunkName: "about" */ '../views/About.vue')
  }
]

const router = new VueRouter({
  mode: 'history',
  base: process.env.BASE_URL,
  routes
})

export default router
```

การเรียกใช้ Router

```

<template>
  <v-app>
    <v-app-bar app>
      <v-toolbar-title class="headline text-uppercase">
        <span>vuetify </span>
        <span class="font-weight-light"> training</span>
      </v-toolbar-title>
      <v-spacer></v-spacer>
      <v-btn
        text
        href="https://github.com/vuetifyjs/vuetify/releases"
        target="_blank"
      >
        <span class="mr-2">button</span>
      </v-btn>
    </v-app-bar>
    <v-content>
      <router-view/>
    </v-content>
  </v-app>
</template>

```

```

<v-content>
  <router-view/>
</v-content>

```


TRY IT !

- สร้าง Login.vue ที่ views

=> การสร้างหน้า Login

- สร้าง Router เพื่อตั้งหน้า Login อยู่ที่
- root path ในไฟล์ index.js ที่ folder router

```
{
  path: '/',
  name: 'login',
  component: Login
},
```

การสร้าง Nest Route

=> เพิ่มที่ไฟล์ router.js

```
{
  path: '',
  name: 'Toolbar',
  component: Toolbar,
  children: [
    {
      path: '/home',
      name: 'home',
      component: Home
    },
    {
      path: '/about',
      name: 'about',
      component: About
    }
  ]
}
```



Data binding

การนำเอาข้อมูลจากตัวแปรมาใช้

```

1  <template>
2  |   <div>
3  |     <h1>Welcome to Login page</h1>
4  |     {{ data }}
5  |   </div>
6  </template>
7
8  <script>
9  export default {
10 |   data: () => ({
11 |     data: 'Login'
12 |   })
13 | }
14 </script>

```

เป็นการแสดงข้อมูลขึ้นมาโชว์
หน้าเว็บไซต์

V-if

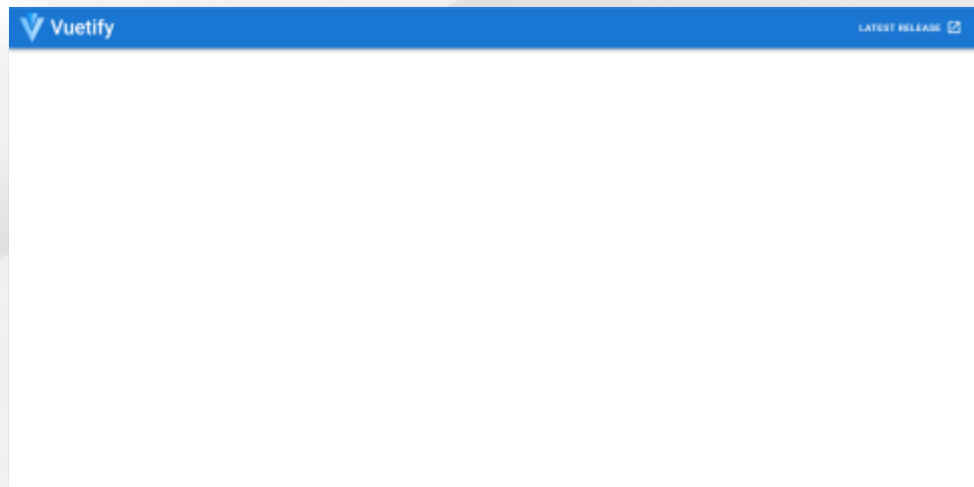
condition

```

<template>
  <h1 v-if="show">{{ data }}</h1>
</template>

<script>
export default {
  data () {
    return {
      data: 'Hello from Login',
      show: false
    }
  }
}
</script>
  
```

Result



V-for

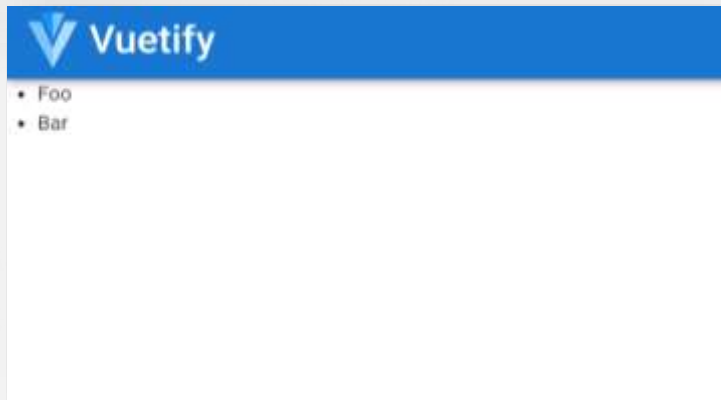
```

<template>
  <div>
    <h1 v-if="show">{{ data }}</h1>
    <ul>
      <li v-for="(item, index) in items" :key="index">
        {{ item.message }}
      </li>
    </ul>
  </div>
</template>

<script>
export default {
  data () {
    return {
      data: 'Hello from Login',
      show: false,
      items: [{ message: 'Foo' }, { message: 'Bar' }]
    }
  }
}
</script>

```

Result



methods: { }

เปลี่ยนแปลงหรือกระทำต่างๆ กับข้อมูล

```
<v-btn color="success" @click="display()">show</v-btn>
```

```
methods: {
  display () {
    alert('Display Function Alert')
  }
}
```



V-model

```

<v-flex xs4>
  <v-text-field
    v-model="email"
    label="E-mail"
    prepend-icon="mdi-email"
    id="email"
  ></v-text-field>
</v-flex>
<v-flex xs4>
  <v-text-field
    v-model="password"
    label="Password"
    prepend-icon="mdi-lock"
    id="password"
  ></v-text-field>
</v-flex>

```

```

<script>
export default {
  data () {
    return {
      email: '',
      password: ''
    }
  },
  methods: {
    signin () {
      console.log(this.email)
    }
  }
}
</script>

```

ให้ทำตัดเกรด โดย รับค่าคะแนน จาก *textfield* แล้วกดปุ่มเพื่อให้แสดงเกรดที่ได้
ตามช่วงคะแนนดังนี้

A 80 - 100

B 70 - 79

C 60 - 69

D 50 - 59

F 0 - 49

ถ้าหากมีการกรอก ตัวอักษร หรือ ตัวเลขที่ไม่อยู่ในช่วง 0 - 100 จะต้องมีแจ้งเตือน

Project Presentation

โจทย์ : ให้ออกแบบระบบแนะนำตัวเอง โดยมีข้อกำหนดดังนี้

1. มีหน้า login
2. เมื่อเข้าหน้า login มาแล้วให้แสดงผลประวัติของตัวเองโดยย่อ
(แสดงแค่ รหัสพนักงาน , ชื่อ-นามสกุล)
แล้วมี ปุ่มแสดงรายละเอียดเป็น dialog ขึ้นมาโชว์ ประวัติแบบเต็ม
- *3. ให้สร้างตารางโดยใช้ข้อมูลจาก database มาแสดงในตาราง
- *4. ข้อมูลที่ได้ สามารถเพิ่ม ลบและแก้ไขข้อมูลได้

```
{  
  employee_id : "",  
  firstname:"",  
  lastname:"",  
  nickname:"",  
  age:"",  
  graduated_from : <university>  
  what_about_me : {  
    hobby:"",  
    favorite_color:"",  
    favorite_food:"",  
    language_programming: [],  
    computer_skill:""  
  }  
}
```

Component Communication การสื่อสารระหว่าง Component

- Props เป็นการสื่อสารแบบบนลงล่างคือ Parent > Child
- Events เป็นการสื่อสารแบบล่างขึ้นบนคือ Child > Parent
- Event Bus เป็นการสื่อสารแบบ Publish/Subscribe

PROPS

การส่งค่าข้าม component

Sub Component

```
<template>
  <div>
    Simple Component
    <br />
    ค่าที่ได้รับ : {{ send_dialog }}
  </div>
</template>

<script>
export default {
  props: ['send_dialog'],
  data () {
    return {}
  }
}
</script>
```

Components/SimpleCom.vue

ปุ่ม click ใช้เรียก
sendData()
เพื่อเปิด dialog

Sub Component

Main Component

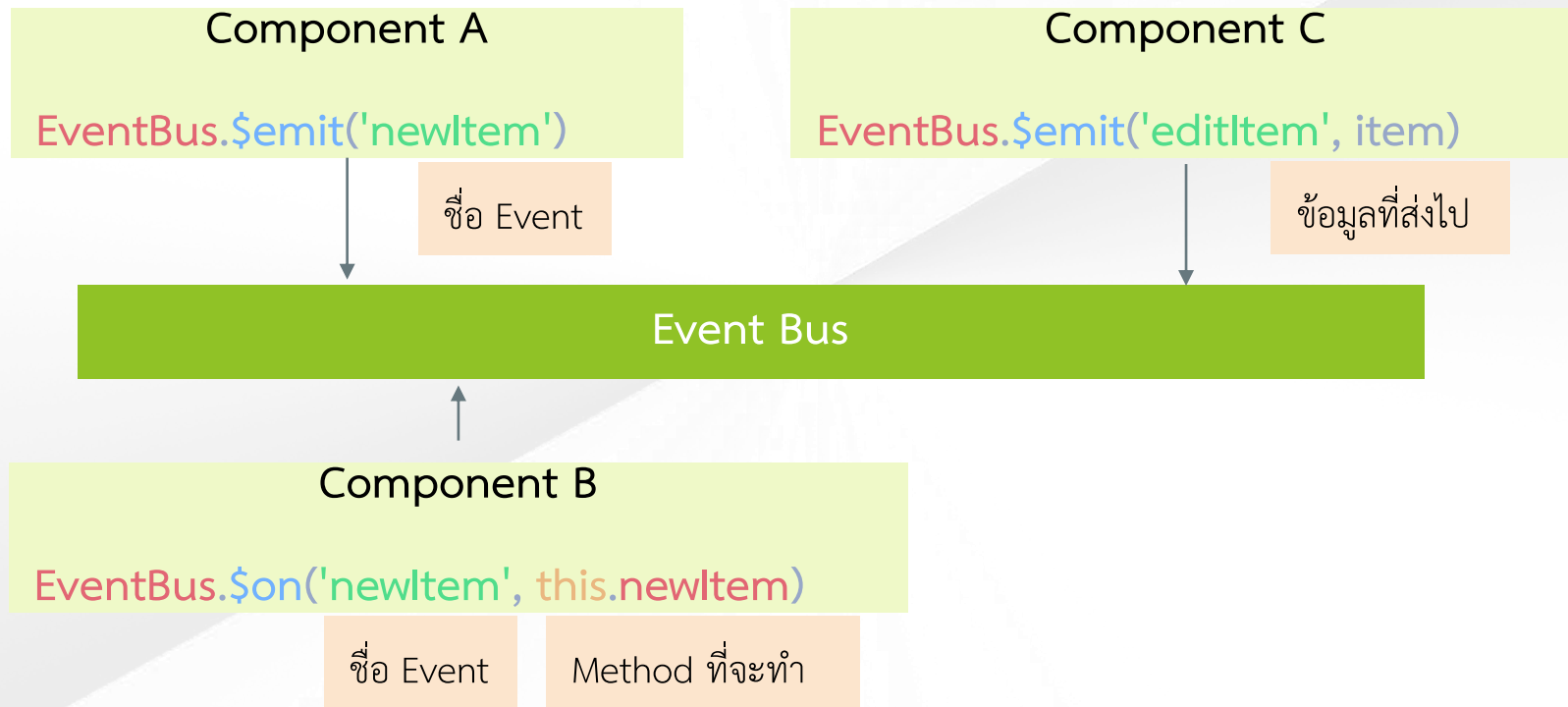
Views/SimpleView.vue

```
<template>
  <div class="pa-5">
    <v-btn color="primary" dark @click="sendData()">
      Click Me </v-btn>
    <v-dialog v-model="dialog" width="500">
      <v-card height="300">
        <Simple-Com :send_dialog="data_send_props" />
      </v-card>
    </v-dialog>
  </div>
</template>

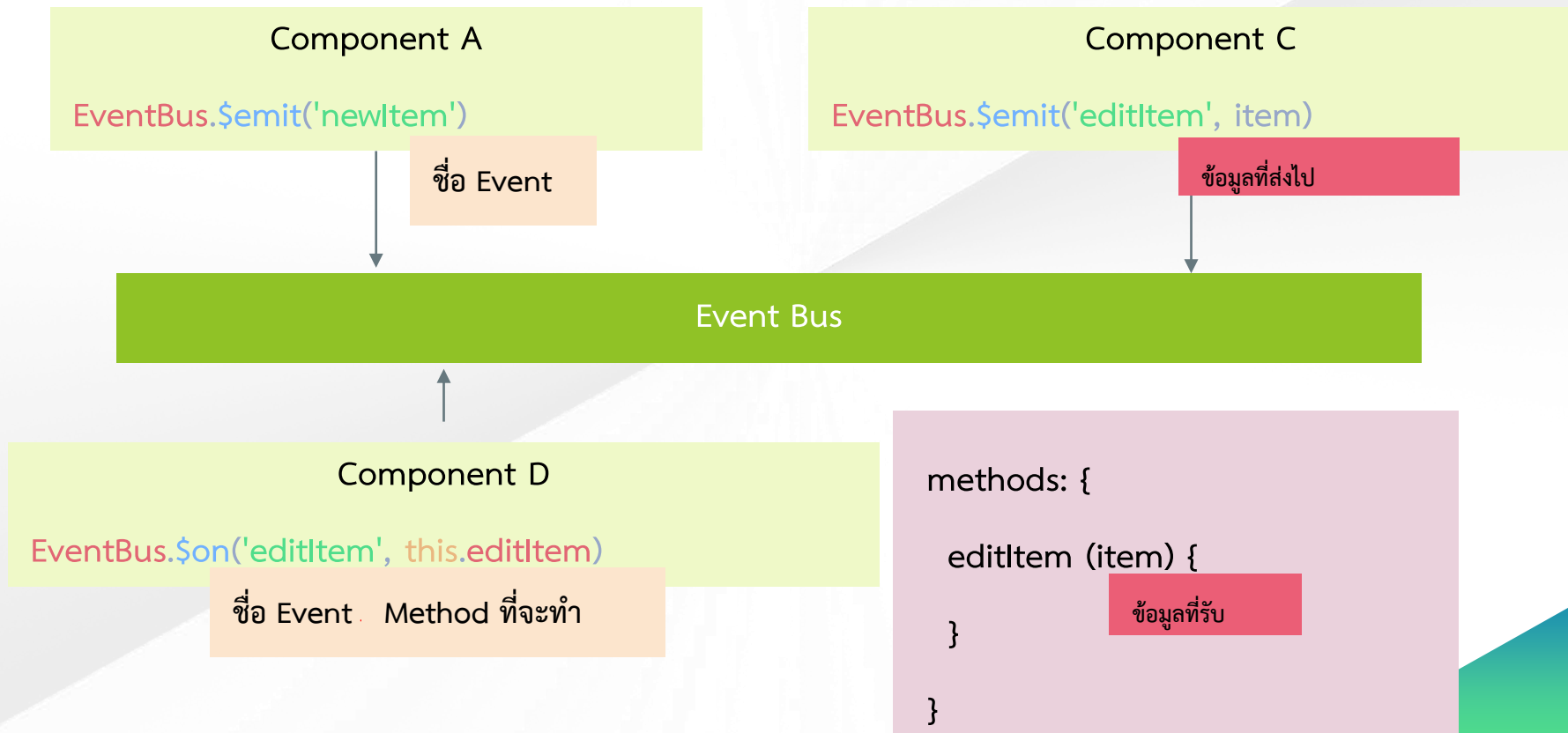
<script>
import SimpleCom from '../components/SimpleCom'
export default {
  components: {
    SimpleCom
  },
  data () {
    return {
      dialog: false,
      data_send_props: 'send value from Simple'
    }
  },
  methods: {
    sendData () {
      this.dialog = true
    }
  }
}
```

ประกาศไว้เพื่อเรียกใช้ค่าที่ props มาจาก component อื่น

Event Bus



Event Bus



Event Bus

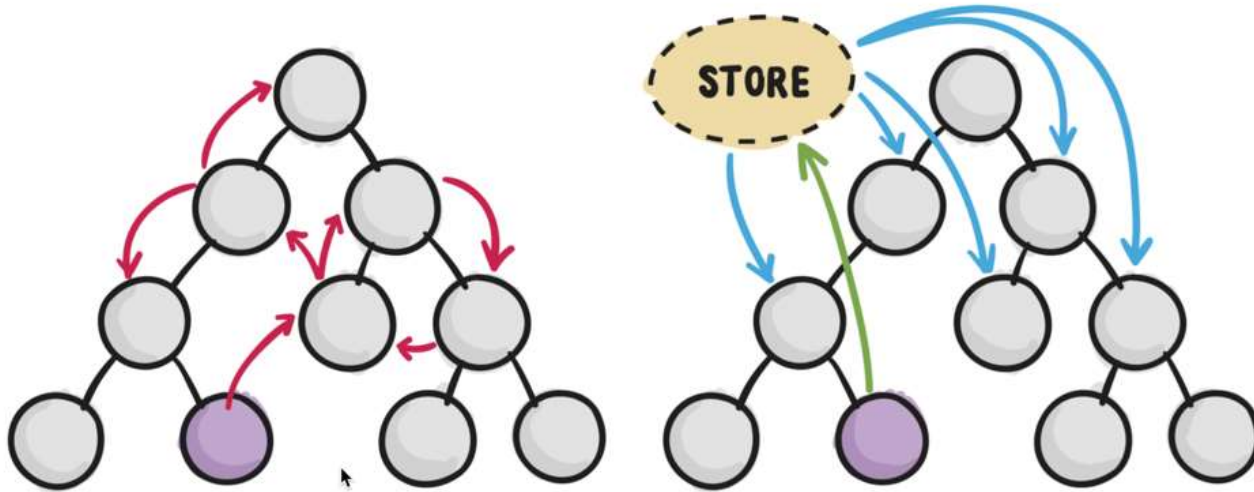
สร้างไฟล์ EventBus.js ที่ folder src

```
import Vue from 'vue'  
export const EventBus = new Vue()
```

การเรียกใช้ จะต้อง import ไว้ทุกหน้าที่จะมีการใช้ EventBus

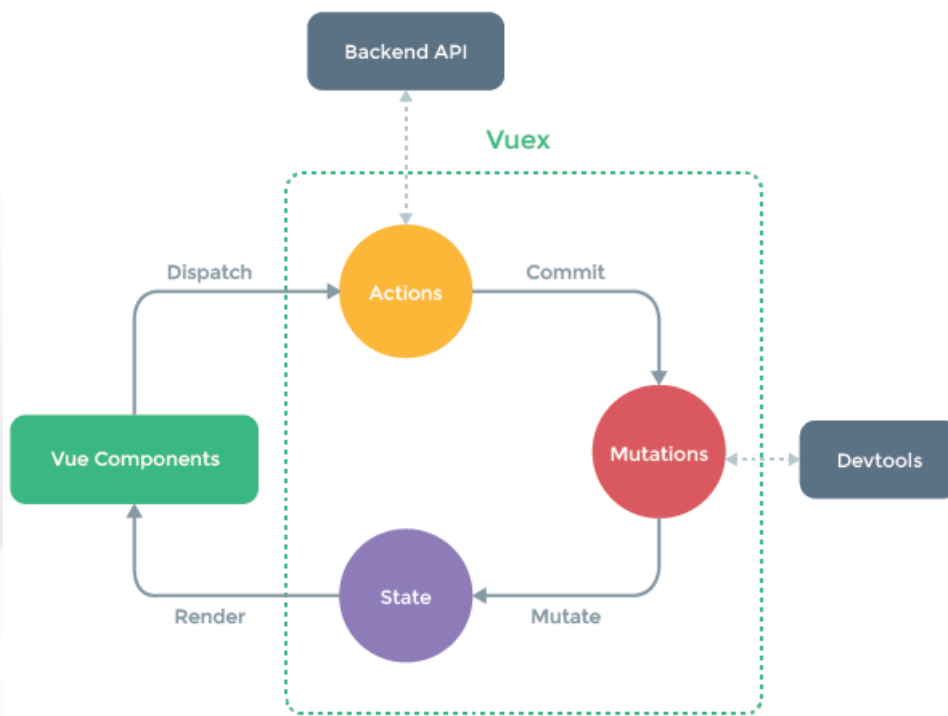
```
import { EventBus } from '@/EventBus'
```

Vuex Vuex-pathify



 COMPONENT INITIATING CHANGE

“**Vuex** is a state management pattern + library for Vue.js applications.” เป็น Library ตัวหนึ่งที่น่าสนใจในโปรเจก Vue.js โดยประโยชน์ของมันคือ การสร้างเป็น Store เพื่อมาจัดการ Data Flow และ State Data ที่อยู่ใน Component ช่วยจัดการ Code ที่ซ้ำซ้อนและจัด Code ให้เป็นระบบมากขึ้น



ส่วนประกอบของ Vuex ใน Store มีอะไรบ้าง

State => ขั้นตอนนี้เป็นขั้นตอนที่เก็บข้อมูลของ Store สามารถที่จะออกแบได้ตามที่ต้องการ ไม่ว่าจะเป็น String , Object

Getters => ขั้นตอนนี้ใช้ในกรณีที่ต้องการนำข้อมูลมาประมวลผลก่อนถูกเรียกไปใช้งาน ซึ่งในส่วนนี้สามารถกำหนดเป็น method ที่เรียกใช้งานบ่อย ๆ ได้

Mutations => เป็นขั้นตอนในการเปลี่ยนแปลงข้อมูลใน State ที่ได้รับจากการ Commit ที่ได้มาจาก Action

Actions => ขั้นตอนนี้เป็นการจัดการ Action ต่าง ๆ เช่น ดึงข้อมูลจากระบบ API หรือการทำ Logic Flow ต่าง ๆ หรือการเก็บข้อมูลเอาไว้ที่ Database ในขั้นตอนนี้สามารถทำ Async/Await ได้ด้วย แล้วจึง Commit ไปยัง Mutation ต่อไป

ตัวอย่างการใช้ Vuex

```
export default new Vuex.Store({
  state: {
    show_value: 150
  },
  getters: {
    show_value(state) {
      return "Value :" + state.show_value
    }
  },
  mutations: {
    change_data(state) {
      state.show_value += 10
    }
  },
  actions: {
    show_data(context) {
      console.log('click')
      context.commit('change_data')
    }
  },
  modules: {
  }
})
```

Store index.js

```
<v-container fluid class="ma-2 pa-4">
  <!-- fluid ใช้สำหรับกรณีที่ต้องการให้ v-container แสดงเต็มพื้นที่ -->
  <v-row> Hello From Vue </v-row>
  <v-row> {{ show_value }}</v-row>
  <v-row>
    <v-btn color="primary" outlined @click="show_data">Click Show</v-btn>
  </v-row>
</v-container>
</template>
```

ต้อง import Vuex มาใช้โดยเขียนรูปแบบนี้

```
import { mapActions, mapGetters } from "vuex";
export default {
  name: "Helloworld",
  data: () => ({}),
  computed: {
    ...mapGetters([
      show_value: "show_value",
    ]),
  },
  methods: {
    ...mapActions([
      show_data: "show_data",
    ]),
  },
};
```

ใช้ map ตัวแปร getters

ใช้ map function Action

ผลแสดงหน้า

เก็บ

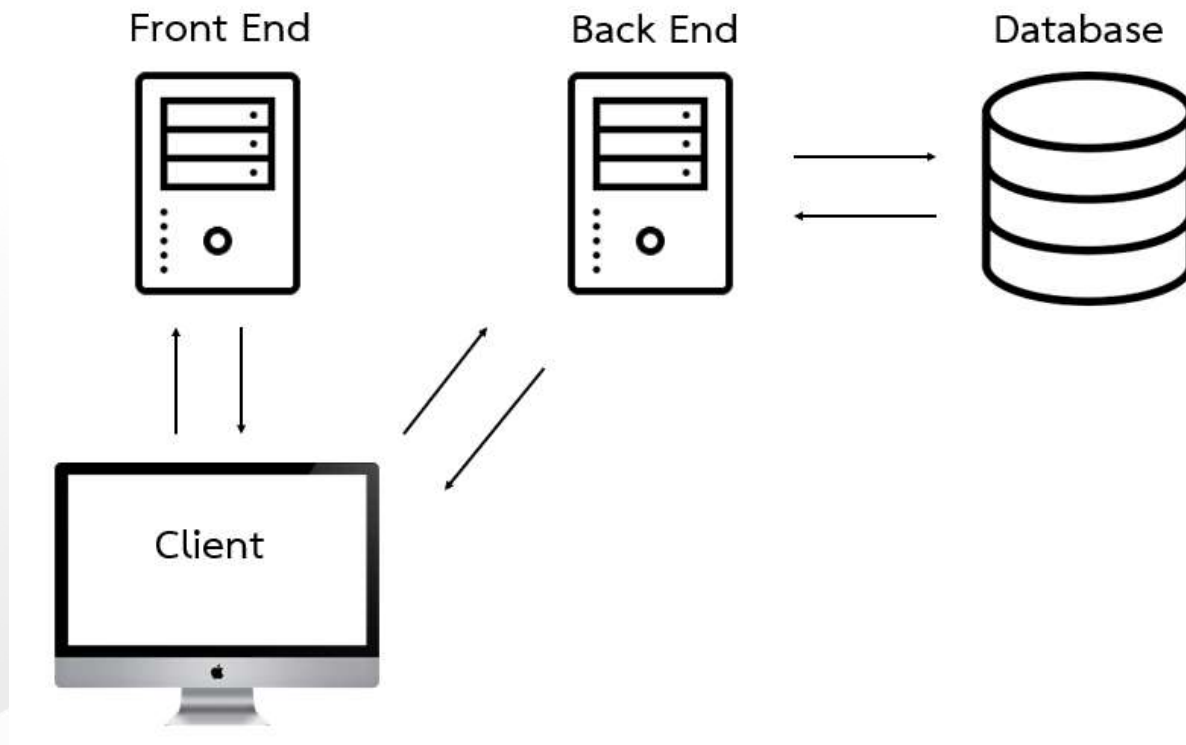
Hello From Vue

Value :150

CLICK SHOW

Component
HelloWorld.vue

โครงสร้างการทำงาน



Axios

คำสั่ง Install

yarn add axios vue-axios หรือ

npm install --save axios vue-axios --force

*กรณี error ให้เพิ่มต่อท้าย - -force

```
import axios from 'axios'  
  
import VueAxios from 'vue-axios'  
  
Vue.use(VueAxios, axios)
```

เพิ่มที่ main.js

รูปแบบการใช้งานของ Axios นิยมใช้

4 แบบหลักๆ ดังนี้

GET - ดึงข้อมูล

POST - สร้างข้อมูล

PUT - อัปเดตข้อมูล

DELETE - ลบข้อมูล

Axios - GET

```
<script>
export default {
  created () {
    this.getData()
  },
  methods: {
    getData () {
      this.axios.get('http://labkk.ga:3000/persons').then((response) => {
        console.log(response.data)
      })
    }
  }
}
</script>
```

Axios – Async & Await

```
getData () {  
  this.axios.get('http://labkk.ga:3000/persons').then((response) => {  
    console.log(response.data)  
  })  
}
```



```
async getData () {  
  try {  
    var {data} = await this.axios.get('http://labkk.ga:3000/persons')  
    console.log(data)  
    this.datas = data  
  } catch (error) {  
    console.log(error.message)  
  }  
}
```

Axios - POST

```
try {  
  var {data} = await this.axios.post('http://labkk.ga:3000/person',  
ข้อมูลที่ต้องการสร้าง)  
  console.log(data)  
  
}catch (error) {  
  console.log(error.message)  
}
```

Axios - PUT

```
try {  
    var {data} = await this.axios.put('http://labkk.ga:3000/person/'+  
    ไอดีที่ต้องการแก้ไข , ข้อมูลใหม่)  
    console.log(data)  
  
} catch (error) {  
    console.log(error.message)  
}
```


Axios - DELETE

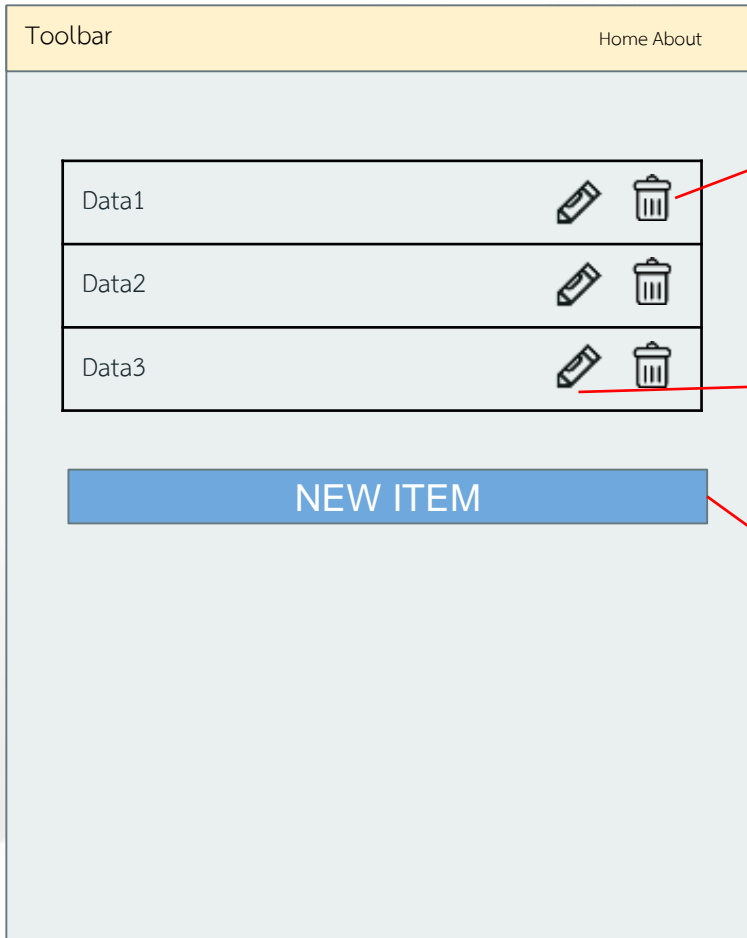
```
try {  
    var { data } = await this.axios.delete('http://labkk.ga:3000/person/' + ไอดีที่ต้องการจะ  
ลบ  
)  
    console.log(data)  
  
} catch (error) {  
    console.log(error.message)  
}
```

Project Presentation

โจทย์ : ให้ออกแบบระบบแนะนำตัวเอง โดยมีข้อกำหนดดังนี้

1. มีหน้า login
2. เมื่อเข้าหน้า login มาแล้วให้แสดงผลประวัติของตัวเองโดยย่อ
(แสดงแค่ รหัสพนักงาน , ชื่อ-นามสกุล)
แล้วมี ปุ่มแสดงรายละเอียดเป็น dialog ขึ้นมาโชว์ ประวัติแบบเต็ม
- *3. ให้สร้างตารางโดยใช้ข้อมูลจาก database มาแสดงในตาราง
- *4. ข้อมูลที่ได้ สามารถเพิ่ม ลบและแก้ไขข้อมูลได้

```
{  
  employee_id : "",  
  firstname:"",  
  lastname:"",  
  nickname:"",  
  age:"",  
  graduated_from : <university>  
  what_about_me : {  
    hobby:"",  
    favorite_color:"",  
    favorite_food:"",  
    language_programming: [],  
    computer_skill:""  
  }  
}
```









Dialog confirm

Show Dialog Edit item

Show Dialog Add item

Toolbar

HomeAbout

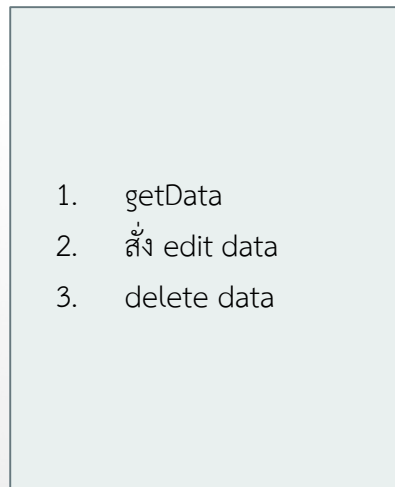
Data1		
Data2		
Data3		

NEW ITEM

Component Dialog

Show Dialog Edit item

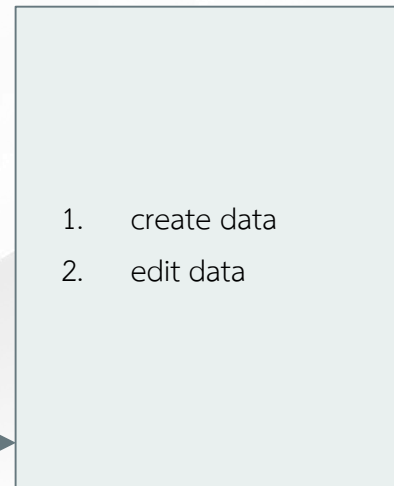
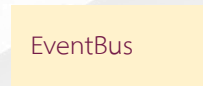
Show Dialog Add item



Component Table



Component AddItem



Component Dialog

