

CIS 658 Web Architectures

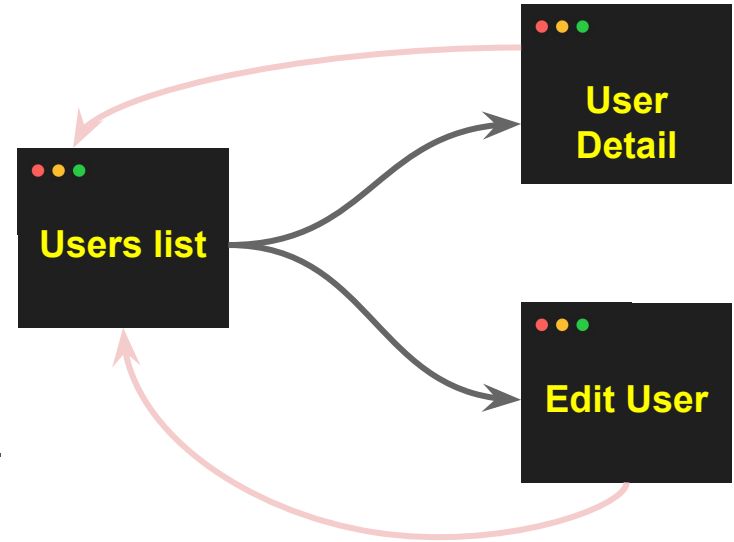
Vuetify



GRAND VALLEY
STATE UNIVERSITY®

Lecturer: **Dr. Yong Zhuang**

Practice: Random User App



Update Home.vue so that:

- Clicking a name opens the related Detail.vue page.
- Clicking Edit opens the related Edit.vue page.

Resources: <https://vuetifyjs.com/en>

Online Playgrounds
<https://play.vuetifyjs.com/>

What is Vuetify?

Material Design focuses on using grid-based layouts, responsive animations and transitions, padding, and depth effects like lighting and shadows.

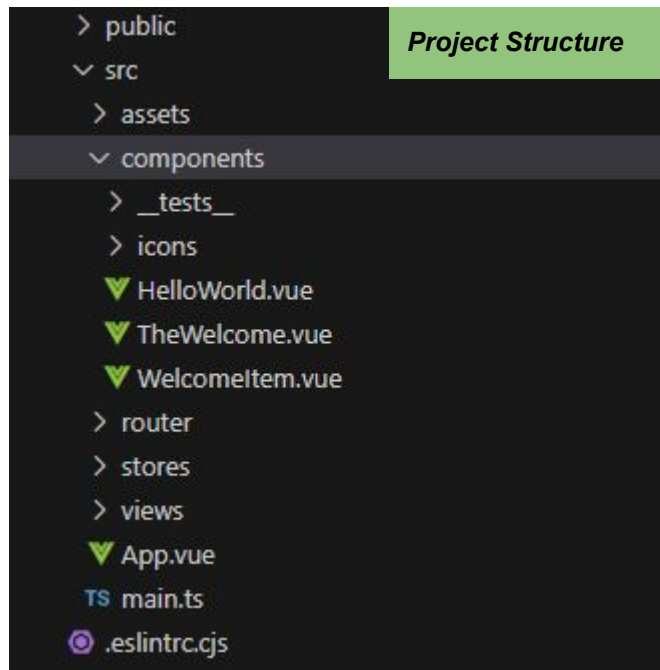
- Overview
 - A **Material Design** Framework for Vue.js
 - Vuetify version 2.6.x does not support Vue 3.x
 - Vuetify version 3.0 (Oct 22) supports Vue 3.0
- Styles, Typography
 - (Material) Color Names as class names
- UI Components
 - Code snippet showing both <template> and <script>
 - Detailed Element API (properties to HTML attributes and VueJS binding)

Why Vuetify? NPM Trends

- It is a powerful Vue component framework
- Easy to learn and easy to master
- The collection of UI components in Vuetify maintains a consistent style throughout the application and provides enough customization options to suit any use case.



Getting Started



```
npm install --save vuetify
npm i vite-plugin-vuetify
```

src/main.ts

```
import { createApp } from 'vue'
import { createVuetify } from 'vuetify'
import App from './App.vue'
import './assets/main.css'
import 'vuetify/styles'
import * as components from 'vuetify/components'
import * as directives from 'vuetify/directives'

const vuetify = createVuetify({ components, directives })
createApp(App).use(vuetify).mount('#app')
```

Using Icon Fonts

```
npm install @mdi/font
npm install @fortawesome/fontawesome-free -D
```

```
<!-- using the icon in your UI -->
<v-icon icon="mdi-cog" />
```

Material Design Icons

```
import '@mdi/font/css/materialdesignicons.css'
import '@fortawesome/fontawesome-free/css/all.css'
import { aliases, mdi } from 'vuetify/iconsets/mdi'
import { fa } from 'vuetify/iconsets/fa'

const vuetify = createVuetify({
  icons: {
    defaultSet: 'mdi',
    aliases,
    sets: {
      mdi,
      fa
    }
  },
  components,
  directives
})

createApp(App).use(vuetify).mount('#app')
```

Vuetify built-in components

- Navigation
 - v-app-bar, v-bottom-navigation, v-footer, v-navigation-drawer, v-tabs, ...
- Containment
 - v-btn, v-card, v-chip, v-dialog, v-expansion-panel, v-list, v-menu, v-toolbar, ...
- Input, Selection, & Controls
 - v-autocomplete, v-checkbox, v-file-input, v-radio, v-select, v-slider, v-switch
 - v-btn-toggle, v-carousel, v-chip-group, v-slide-group
- Grid system
- Feedback
 - v-alert, v-badge, v-snackbar, v-rating, ...

Vuetify Element Names

TitleCase	kebab-case
VAlert	v-alert
VNavigationDrawer	v-navigation-drawer

Either style will work, but use them consistently throughout the code

Other UI Framework Alternatives

Framework	Component Names
Element UI	<el-xxxx>
Bootstrap Vue	<b-xxxx>
Quasar	<q-xxxx>
Iconic	<ic-xxxx>

- *These frameworks provide **components***
- *Both the **visual** style and **inner logic** (to some extent) of these components are customizable*

Framework	Component Names
Bootstrap	N/A
Tailwind CSS	N/A

- *These frameworks provide **CSS classes** ONLY for visual styling*

Navigation Components

Typical Application Layout

```
<template>
  <v-app>
    <v-app-bar>
      <span>Toolbar</span>
    </v-app-bar>
    <v-main>
      <!-- UI content goes here -->
    </v-main>
    <v-footer app>
      <span>Footer</span>
    </v-footer>
  </v-app>
</template>
```

Toolbar

Footer

Important: be sure to include the `app` attribute on `v-footer` to pin it the application layout!

<v-app-bar>

```
<template>
  <v-app>
    <v-app-bar app>
      <v-app-bar-title>CS371 WebAppDev</v-app-bar-title>
      <v-spacer></v-spacer>
      <v-icon color="yellow" icon="mdi-home" />
      <v-icon color="yellow">mdi-exit-to-app</v-icon>
    </v-app-bar>
    <v-footer app color="grey">
      <span>Footer</span>
    </v-footer>
  </v-app>
</template>
```

Material icon Name

Vuetify Demo

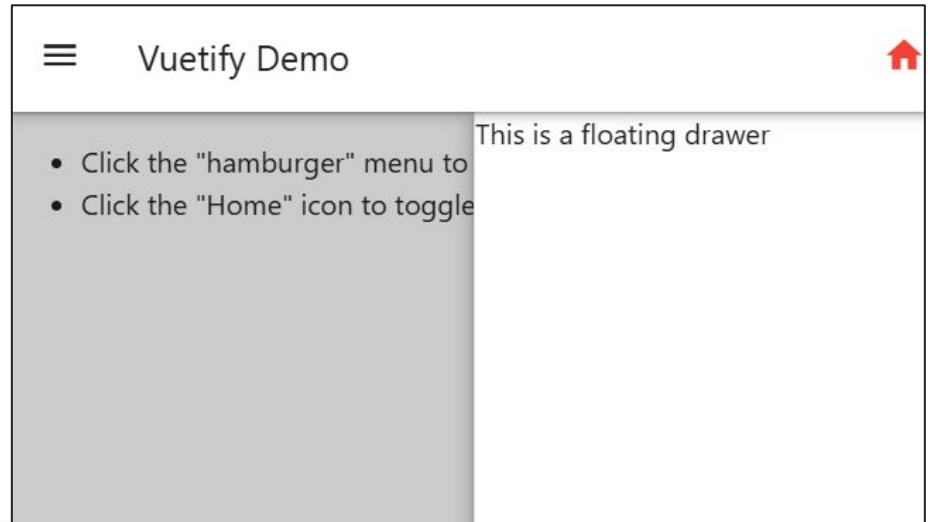


Footer

Demo

Navigation Drawer

- `<v-navigation-drawer>` is a direct child of `<v-app>`
- Default: floating navigation drawer. When unhidden, the navigation drawer will float above the main content
- Permanent floating navigation drawer: when unhidden, the navigation drawer will push contents aside



Demo

Using Color Class Names

Softer variations – good for backgrounds, sections, and layout structure

Color Usage	Class name	Example
Background Color	bg-{color}	<code><div class="bg-orange"> ... </div></code>
Text Color	text-{color}	<code><p class="text-green" />...</p></code>
Darker	bg-{color}-darken-{n} text-{color}-darken-{n}	<code><div class="bg-orange-darken-1"> ... </div></code>
Lighter	bg-{color}-lighten-{n} text-{color}-lighten-{n}	<code><div class="bg-orange-lighten-1"> ... </div></code>
Accent	bg-{color}-accent-{n} text-{color}-accent-{n}	<code><div class="bg-orange-accent-1"> ... </div></code>

1~4

1~5

1~4

Flashier, more attention-grabbing – good for emphasis, highlights, buttons

Demo

Input, Selection, and Controls

User Inputs

- v-autocomplete / v-combobox / v-select
- v-checkbox / v-switch
- v-radio-group and v-radio
- v-slider / v-range-slider
- v-form & v-text-field & v-textarea
- v-date-picker
- v-time-picker
- v-menu

Important attribute: v-model

Input Field Validation

```
<template>
  <v-text-field label="Email" :rules="eRules" />
</template>
```

```
<script setup lang="ts">
const eRules = [
  function (x: string): boolean | string {
    if (x.indexOf('@') >= 1) return true
    else return 'Invalid email format'
  },
  function (x: string): boolean | string {
    if (x.endsWith('.edu') || x.endsWith('.org') || x.endsWith('.net')) return true
    else return 'Only .edu/.org/.net accepted'
  }
]
</script>
```

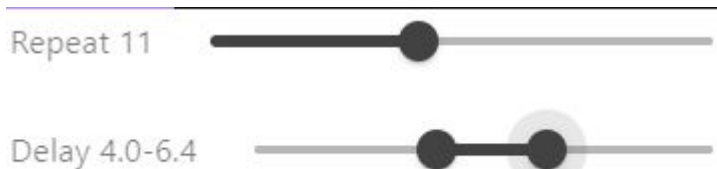
Demo

<v-slider> & <v-range-slider>

```
<template>
  <v-slider v-model="repeat" color="red" :label="`Repeat ${repeat}`" min="1" max="5" />
  <v-range-slider v-model="delay" :label="`Delay ${range}`" step="0.1" min="2" max="5" />
</template>
```

```
<script setup lang="ts">
import { ref, computed } from 'vue'
const repeat = ref(0)
const delay = ref([2.7, 4.3])
const range = computed(() => {
  return delay.value[0].toFixed(1) + '-' + delay.value[1].toFixed(1) + 'secs'
})
</script>
```

Demo



Date & Time Pickers

```
<template>
  <v-app>
    <v-main>
      <v-date-picker v-model="selDate" width="200px" />
    </v-main>
  </v-app>
</template>
```

```
<script setup lang="ts">
  import { ref } from 'vue'
  const selDate = ref(null) // For storing the selected date
</script>
```

SELECT DATE

Enter date

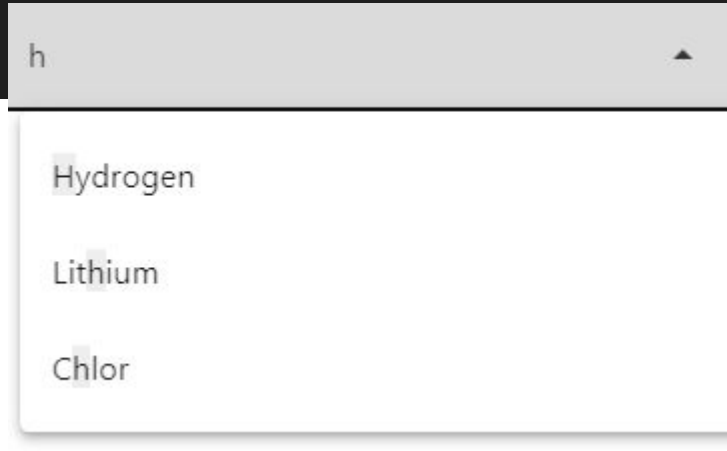
November 2023 < >

S	M	T	W	T	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

Demo

<v-autocomplete> / <v-combobox> / <v-select>

```
<template>
  <v-app>
    <v-main>
      <p>Select your input below</p>
      <v-autocomplete :items="atoms" v-mode="myAtom" />
    </v-main>
  </v-app>
</template>
```



h

- Hydrogen
- Lithium
- Chlor

```
<script setup lang="ts">
  import { ref } from 'vue'

  const myAtom = ref('')
  const atoms = ref([
    'Hydrogen',
    'Lithium',
    'Sodium',
    'Pottasium',
    'Rubidium',
    'Chlor',
    'Carbon',
    'Tin',
    'Lead',
    'Gold',
    'Copper'
  ])
</script>
```

Demo

<v-radio-group> & <v-radio>

```
<template>
  <v-app>
    <v-main>
      <v-radio-group label="Tax Filing Status" v-model="taxStatus">
        <v-radio v-for="(s, opt) in taxFiling" :label="s" :value="opt" :key="s" />
      </v-radio-group>

      <p>Your selection is {{ taxStatus }}</p>
    </v-main>
  </v-app>
</template>
```

```
<script setup lang="ts">
  import { ref } from 'vue'

  const taxStatus = ref(-1)
  const taxFiling = ref([
    'Single',
    'Married Filing Jointly',
    'Married Filing Separately',
    'Head of Household'
  ])
</script>
```

Tax Filing Status

- Single
- Married Filing Jointly
- Married Filing Separately
- Head of Household

Demo

<v-checkbox> / <v-switch>

```
<template>
  <v-app>
    <v-main>
      <v-checkbox v-model="lazyDel" label="Lazy Delete"> </v-checkbox>
      <v-switch v-model="useWifi" label="Wifi"> </v-switch>
    </v-main>
  </v-app>
</template>
```

Lazy Delete

Wifi

```
<script setup lang="ts">
  import { ref } from 'vue'

  const lazyDel = ref(false)
  const useWifi = ref(true)
</script>
```

Demo

Practice: User Registration Form

Requirements

- Add a v-text-field for the user's name.
- Add a v-select for the user's preferred programming language, with options:
 - JavaScript
 - Python
 - Java
- Add a v-radio-group to let the user choose their experience level:
 - Beginner
 - Intermediate
 - Advanced

