

MỤC TIÊU:

Kết thúc bài thực hành này bạn có khả năng thao tác với:

- ✓ Listen to Events (Lắng nghe sự kiện), Cơ chế phát ra sự kiện
- ✓ Trình xử lý phương thức
- ✓ Các sự kiện về chuột, bàn phím...
- ✓ Cách thức liên kết dữ liệu trong Form binding
- ✓ Sử dụng v-model kết hợp các thuộc tính, giá trị
- ✓ Cơ chế giao tiếp giữa các component

NỘI DUNG

PHẦN I: Event Handling

Bài 1 (2 điểm): Tạo ứng dụng quản lý công việc. Yêu cầu:

- ✚ Tạo form thêm mới một công việc
- ✚ Hiển thị danh sách các công việc, có nút Xóa công việc
- ✚ Có thể sử dụng sự kiện **@submit.prevent** thay vì **@click** ngăn chặn hành vi tải lại trang web khi người dùng bấm 'Thêm công việc'

Quản lý công việc

Tên công việc:

Thêm công việc

Ăn sáng	Xóa
Đi học	Xóa
Chơi bóng rổ	Xóa

Hướng dẫn tham khảo:

```
<template>
  <div class="col-sm-4 p-5">
    <h3 class="text-center">Quản lý công việc</h3>

    <!-- Form thêm công việc -->
    <form @submit.prevent="addList">
      <div class="mb-3">
        <label class="form-label">Tên công việc:</label>
        <input type="text" class="form-control" v-model="newToDo" placeholder="Nhập tên công việc">
      </div>
      <button type="submit" class="btn btn-primary">Thêm công việc</button>
    </form>

    <!-- Danh sách công việc -->
    <ul class="list-group mt-4">
      <li class="list-group-item d-flex justify-content-between align-items-center" v-for="(job, index) in jobs"
        :key="index">
        {{ job }}
        <button class="btn btn-danger btn-sm" @click="removeList(index)">Xóa</button>
      </li>
    </ul>
  </div>
</template>

<script setup>
import { ref } from 'vue';

const newToDo = ref('');
const jobs = ref(['Ăn sáng', 'Đi học', 'Chơi bóng rổ']);

const addList = () => {
  if (newToDo.value.trim()) { //loại bỏ khoảng trống ở đầu và cuối của chuỗi
    jobs.value.push(newToDo.value.trim());
    newToDo.value = ''; // Reset input field
  }
};

const removeList = (index) => {
  jobs.value.splice(index, 1);
};
</script>
```

Bài 2 (3 điểm): Tạo một Form đăng nhập gồm email và mật khẩu

Yêu cầu:

- Form Validation: Sử dụng **v-if** kiểm tra hợp lệ khi người dùng bỏ trống hoặc nhập sai định dạng email/mật khẩu
- Khi người dùng đăng nhập thành công, hiển thị Chào mừng và xử lý thêm chức năng Đăng xuất. Khi bấm Đăng xuất hiển thị lại giao diện Form Đăng nhập
- Sự kiện click trên nút Đăng nhập: Trong Vue.js, khi sử dụng nút bên trong form, sự kiện submit mặc định của form sẽ được kích hoạt. Điều này có thể làm trang web tải lại, do đó bạn cần ngăn chặn hành vi mặc định này bằng **@submit.prevent** thay vì **@click**.

Form Đăng nhập

Email:

Mật khẩu:

Đăng nhập



Chào mừng, teonv@gmail.com!

Đăng xuất

Gợi ý tham khảo:

```

<template>
  <div v-if="!isLoggedIn" class="p-5 col-sm-4">
    <h3>Form Đăng nhập</h3>
    <form @submit.prevent="login">
      <div class="mb-3 mt-3">
        <label>Email:</label>
        <input type="email" class="form-control" v-model="email"
placeholder="Nhập email">
        <p v-if="emailError" style="color: red;">{{ emailError }}</p>
      </div>
      <div class="mb-3">
        <label>Mật khẩu:</label>
        <input type="password" class="form-control" v-model="password"
placeholder="Nhập mật khẩu">
        <p v-if="passwordError" style="color: red;">{{ passwordError }}</p>
      </div>
      <button type="submit" class="btn btn-primary">Đăng nhập</button>
    </form>
  </div>

  <div v-else class="p-5 col-sm-5">
    <h3>Chào mừng, {{ email }}!</h3>
    <button @click="logout" class="btn btn-primary">Đăng xuất</button>
  </div>
</template>
<script setup>
import { ref } from 'vue';

const isLoggedIn = ref(false);
const email = ref('');
const password = ref('');

const emailError = ref('');

```

```
const passwordError = ref('');

const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;

const login = () => {
  // Reset thông điệp lỗi
  emailError.value = '';
  passwordError.value = '';

  // Validate email
  if (!email.value) {
    emailError.value = 'Email là bắt buộc.';
  } else if (!emailRegex.test(email.value)) {
    emailError.value = 'Vui lòng nhập email hợp lệ.';
  }





  // Validate mật khẩu
  if (!password.value) {
    passwordError.value = 'Mật khẩu là bắt buộc.';
  }

  // Nếu không có lỗi, xử lý login
  if (!emailError.value && !passwordError.value) {
    isLoggedIn.value = true;
  }
}

const logout = () => {
  isLoggedIn.value = false;
  email.value = '';
  password.value = '';
  emailError.value = '';
  passwordError.value = '';
}
</script>
```

PHẦN II: Form binding trong VueJS

Bài 3 (2 điểm): Tạo form đăng ký người dùng. Yêu cầu như sau:

-  Có các trường: Họ tên, email, Mật khẩu, Ngày sinh, Giới tính, Ngôn ngữ
-  Cho phép người dùng nhấn vào nút Đăng ký.
-  Hiển thị các thông tin đã đăng ký (trừ Mật khẩu).
-  Ứng dụng bài học về phần Binding đơn giản

Form Đăng Ký

Họ tên:

Tèo

Email:

tien@gmail.com

Mật khẩu:

...

Ngày sinh:

30/08/2024


Giới tính: ☒ Nam ☐ Nữ ☐ Khác

Ngôn ngữ: ☒ Tiếng Việt ☒ Tiếng Anh ☐ Tiếng Nhật

Đăng ký

Thông tin đã đăng ký:

Họ tên: Tèo

Email: tien@gmail.com

Ngày sinh: 2024-08-30

Giới tính: Nam

Ngôn ngữ: Tiếng Việt, Tiếng Anh

Bài 4 (3 điểm): Tạo ứng dụng bình luận bài viết. Yêu cầu gồm 3 component:

- Component **Đăng nhập**: Chứa form đăng nhập với tên và mật khẩu.
- Component **Bình Luận**: Cho phép người dùng đã đăng nhập bình luận bài viết.
- Component Chính (**Bai4.vue**): Điều khiển logic hiển thị và luân chuyển giữa các component
- Ứng dụng kiến thức Cơ chế giao tiếp giữa các component thông qua **props** và **emit**

Đăng Nhập

Tên đăng nhập:

Binh An

Mật khẩu:

...

Đăng nhập

Bình luận bài viết



8 loại rau củ quả giàu canxi

Canxi là khoáng chất cần thiết đối với cơ thể người. Có nhiều cách để bổ sung canxi, trong đó bổ sung qua đường ăn uống là cách tốt nhất. Có 8 loại rau củ quả giàu canxi...

Nhập bình luận của bạn

Gửi bình luận

Danh sách các bình luận:

- Binh An**: Bài viết rất hữu ích

Hướng dẫn:

1. Tạo component **LoginComponent.vue** chứa form đăng nhập gồm tên và mật khẩu.

```
<template>
  <div class="m-5 col-sm-3">
    <h2>Đăng Nhập</h2>
    <form @submit.prevent="handleLogin">
      <div class="mb-3">
        <label for="username">Tên đăng nhập:</label>
        <input type="text" class="form-control" id="username" v-model="username"
          placeholder="Nhập tên đăng nhập" />
      </div>

      <div class="mb-3">
        <label for="password">Mật khẩu:</label>
        <input type="password" class="form-control" id="password" v-model="password"
          placeholder="Nhập mật khẩu" />
      </div>

      <button type="submit" class="btn btn-primary">Đăng nhập</button>
    </form>
  </div>
</template>

<script setup>
import { ref } from 'vue';

// Khai báo biến lưu dữ liệu form đăng nhập
const username = ref('');
const password = ref('');

// Hàm xử lý đăng nhập
const emit = defineEmits(['loggedIn']);
function handleLogin() {
  if (username.value && password.value) {
    // phát sự kiện 'loggedIn'
    emit('loggedIn', username.value);
  }
}
</script>
```

2. Tạo component **CommentComponent.vue** chứa giao diện phép người dùng đã đăng nhập bình luận bài viết.

```
<template>
  <div class="col-sm-4 m-5">
    <h2>Bình luận bài viết</h2>
    <div class="card">
      
      <div class="card-body">
        <h3 class="card-title">8 loại rau củ quả giàu canxi</h3>
        <p class="card-text">Canxi là khoáng chất cần thiết đối với cơ thể người. Có nhiều cách để bổ sung canxi, trong đó bổ sung qua đường ăn uống là cách tốt nhất. Có 8 loại rau củ giàu canxi...</p>
      </div>
    </div>
    <form @submit.prevent="submitComment">
      <div class="mt-3">
        <textarea id="commentText" cols="60" v-model="commentText"
          placeholder="Nhập bình luận của bạn"></textarea>
        <button type="submit" class="btn btn-success">Gửi bình luận</button>
      </div>
      <div v-if="comments.length" class="mt-3">
        <h5>Danh sách các bình luận:</h5>
        <ul style="list-style-type: circle;">
          <li v-for="(comment, index) in comments" :key="index">
            <p><strong>{{ comment.name }}</strong>: {{ comment.text }}</p>
          </li>
        </ul>
      </div>
    </div>
  </template>
```

```
<script setup>
import { ref } from 'vue';
const props = defineProps(['username']);

const commentText = ref('');

// Mảng chứa các bình luận đã gửi
const comments = ref([]);

// Xử lý gửi bình luận
function submitComment() {
  if (commentText.value) {
    // Thêm bình luận mới vào mảng comments
    comments.value.push({
      name: props.username, // Sử dụng tên từ props
      text: commentText.value
    });

    // Xóa dữ liệu trong form sau khi gửi
    commentText.value = '';
  }
}
</script>
```

- Điều khiển logic hiển thị và luân chuyển giữa các component **LoginComponent.vue** và **CommentComponent.vue**

```
<template>
  <div>
    <!-- Nếu người dùng chưa đăng nhập, hiển thị form đăng nhập -->
    <LoginComponent v-if="!isLoggedIn" @loggedIn="handleLoginSuccess" />

    <!-- Nếu người dùng đã đăng nhập, hiển thị component bình luận -->
    <CommentComponent v-else :username="loggedInUser" />
  </div>
</template>

<script setup>
import { ref } from 'vue';
import LoginComponent from './LoginComponent.vue';
import CommentComponent from './CommentComponent.vue';

// Quản lý trạng thái đăng nhập và tên người dùng đã đăng nhập
const isLoggedIn = ref(false);
const loggedInUser = ref('');

// Xử lý khi người dùng đăng nhập thành công
function handleLoginSuccess(username) {
  loggedInUser.value = username;
  isLoggedIn.value = true;
}
</script>
```

**** Giảng viên có thể cho thêm các bài tập phù hợp phục vụ assignment.**

***** Yêu cầu nộp bài:**

SV nén file (hoặc share thư mục google drive) bao gồm các yêu cầu đã thực hiện trên, nộp LMS đúng thời gian quy định của giảng viên. **KHÔNG NỘP BÀI COI NHƯ KHÔNG CÓ ĐIỂM.**

--- Hết ---