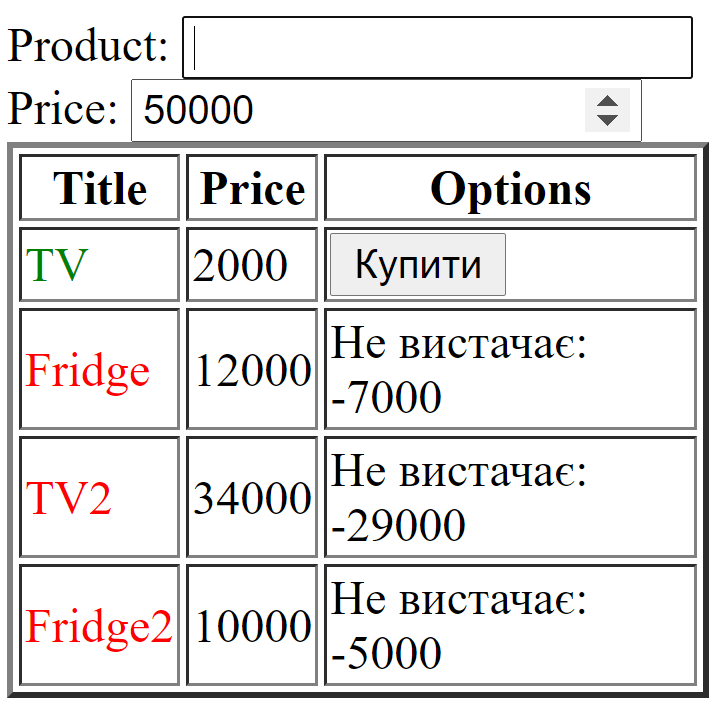
Приклад. Відобразити список товарів. У залежності від кількості грошей користувача (передається у компонент) для товару або відобразити кнопку «купити», або відобразити скільки не вистачає для покупки цього товару. Організувати фільтрацію за назвою та ціною (відображати ті, які менші за задану ціну).



|  |  |
| --- | --- |
| Компонент | <template>  <div>      <label >          Product:          <input type="text" v-model="searchProduct">      </label>      <label >          Price:          <input type="number" v-model="searchPrice">      </label>      <table border="2px">          <tr>              <th>Title</th>              <th>Price</th>              <th>Options</th>          </tr>          <tr v-for="product in filteredProducts" :key="product.id">              <td :class="getProductClass(product)" >                  {{product.title}}</td>              <td>{{product.price}}</td>              <td>                  <span v-if="product.price<=userMoney"><button>Купити</button> </span>                  <span v-else>Не вистачає: {{userMoney-product.price}} </span>                  <!-- <span v-show="product.price<=userMoney"><button>Купити</button> </span> -->                  <!-- <span v-else> {{`Не вистачає: ${userMoney-product.price}`}} </span> -->              </td>          </tr>      </table>      </div>  </template>  <script>      export default {          name:'ProductTable',          props: {              userMoney: {                  type: Number,                  default: 0              },              products: {                  type: Array,                  default: ()=>[]              },          },          data() {              return {                  searchProduct: null,  //Змінна де зберігаємо назву товару, що шукаємо                  filteredProducts:this.products,                  searchPrice: null              }          },          watch: {  //Розділ, у якому записуємо змінні, зміну яких відслідковуємо              searchProduct() { //Функція, яка буде викликатися коли значення "searchProduct" буде змінено                          // "newValue" - нове значення змінної "searchProduct"                  this.filterProducts()              },              searchPrice() {                  this.filterProducts()              }          },          methods: {              getProductClass(product) {                  return product.price<=this.userMoney?'can-buy':'can-not-buy'              },              filterProducts() {                  const val=this.searchProduct.toLowerCase()                    this.filteredProducts=this.products.filter(product=>(                      (!val || product.title.toLowerCase().startsWith(val)) &&                      (!this.searchPrice || product.price<=parseFloat(this.searchPrice))))              }          },      }  </script>  <style lang="css" scoped>  .can-buy{      color:green  }  .can-not-buy{      color:red  }  </style> |
| Використання компонента | <template>    <div>      <!-- 3.Використовуємо компонент, як новий тег -->     <product-table       :user-money="5000"       :products="dataProductsList"     />    </div>  </template>  <script>  //1. Імпортуємо файл  import ProductTable from "./components/ProductTable";  export default {    name: 'App',    components: {    //2.Реєстрація    ProductTable    },    data() {      return {        dataProductsList: [         {           id:1,           title:'TV',           price:2000         },         {           id:2,           title:'Fridge',           price:12000         },         {           id:3,           title:'TV2',           price:34000         },         {           id:4,           title:'Fridge2',           price:10000         },          ]      }    },  }  </script>  <style>  #app {    font-family: Avenir, Helvetica, Arial, sans-serif;    -webkit-font-smoothing: antialiased;    -moz-osx-font-smoothing: grayscale;    text-align: center;    color: #2c3e50;    margin-top: 60px;  }  </style> |