Сучасні каркаси web-додатків

Маршрутизація у Vue Router

Сторінка товару, вкладені розділи та хлібні крихти

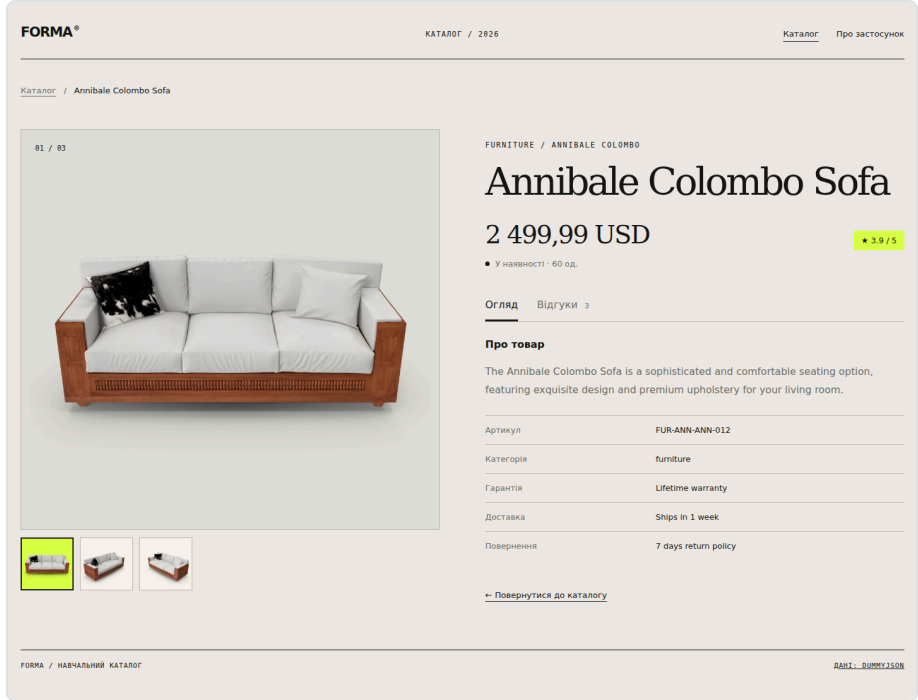
Лекція 6 · Лісняк Андрій Олександрович

Маршрутизація в односторінковому застосунку

Поняття	Зміст
SPA	Один завантажений документ, у якому JavaScript змінює інтерфейс.
Маршрутизація	Зіставлення URL із компонентами представлення.
Запис маршруту	Шлях, компонент і налаштування одного рівня навігації.
Vue Router	Узгоджує представлення, URL та історію браузера.

Каталог і окрема сторінка товару

1. **Каталог** Пошук, картки та дозавантаження з роботи 5.
2. **Товар** Назва, зображення, ціна й наявність за власним URL.
3. **Розділи** Огляд і відгуки всередині спільної оболонки.
4. **Крихти** Посилання на батьківські рівні навігації.

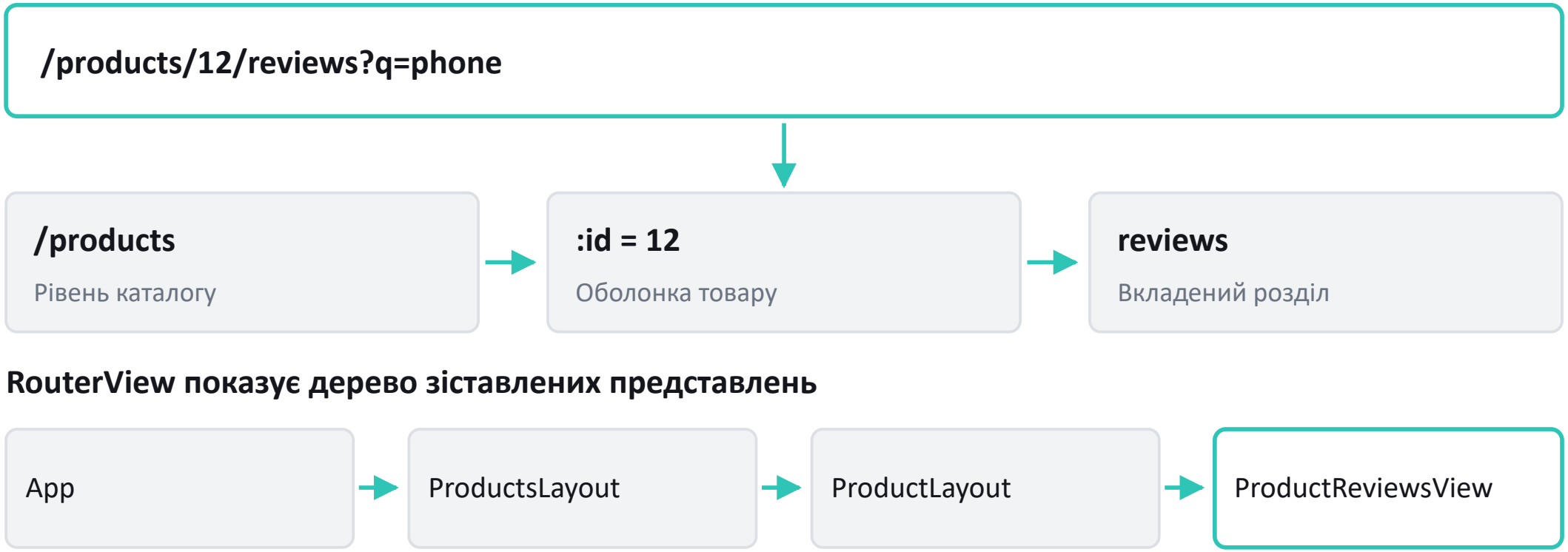


Робочий приклад FORMA з даними DummyJSON.

Шлях, параметр і пошук у URL

Фрагмент	Значення
/products	Рівень каталогу.
12	Значення params.id у динамічному сегменті :id.
/reviews	Розділ відгуків усередині сторінки товару.
?q=phone	Умова пошуку, з якої відкрито товар.

Адреса та дерево представлень



Один URL зіставляється з кількома вкладеними записами маршруту.

Створення маршрутизатора

index.js

```
import { createRouter, createWebHistory } from 'vue-router'
import { routes } from './routes.js'
export const router = createRouter({
  history: createWebHistory(import.meta.env.BASE_URL), routes,
  scrollBehavior(to, from, savedPosition) {
    if (savedPosition) return savedPosition
    if (to.path !== from.path) return { top: 0 }
  },
})
```

Підключення до застосунку

main.js

```
import { createApp } from 'vue'
import App from './App.vue'
import { router } from './router/index.js'
import './styles/main.css'
import './styles/router.css'
createApp(App).use(router).mount('#app')
```

RouterLink і RouterView в оболонці

App.vue · рядки 18–23

```
      <nav class="site-nav" aria-label="Основна навігація">
        <RouterLink to="/products">Каталог</RouterLink>
        <RouterLink :to="{ name: 'about' }">Про застосунок</RouterLink>
      </nav>
    </header>
    <main id="page-content"><RouterView /></main>
```

Адреси застосунку FORMA

Адреса	Представлення
/	Перенаправлення на каталог.
/products	Список товарів та пошук через q.
/products/:id	Оболонка товару й огляд.
/products/:id/reviews	Оболонка товару й відгуки.
/about	Інформація про застосунок.
Невідомий шлях	Окреме представлення NotFoundView.

Команди навігації та історія

Поняття	Зміст
RouterLink / push()	Перехід до нової сторінки додає запис історії.
replace()	Оновлення пошуку замінює поточний запис.
back() / forward()	Рух за записами історії браузера.
Іменована ціль	name, params і query визначають URL переходу.

Параметр id і запит товару

ProductLayout.vue · рядки 7–12

```
const route = useRoute()
const validId = computed(() => /^[1-9]\d*$/.test(String(route.params.id)))
const url = computed(() => validId.value
  ? `https://dummyjson.com/products/${route.params.id}?delay=800` : null)
const { data: product, error, isLoading, refetch } = useFetch(url)
const isMissing = computed(() => !validId.value || error.value?.status === 404)
```

Новий id у тому самому компоненті

1. **Початковий маршрут** ProductLayout показує товар 12.
2. **Перехід** URL змінюється на адресу іншого товару.
3. **Повторне використання** Оболонка може залишитися тим самим екземпляром.
4. **Реакція** Зміна params.id оновлює URL API та запускає запит.

Новий запит очищує попередній стан

useFetch.js · рядки 17–30

```
  watchEffect(async (onCleanup) => {
    refreshKey.value
    const requestUrl = toValue(url)
    activeController?.abort()
    activeController = undefined
    data.value = null
    error.value = null
    isLoading.value = false
    if (!requestUrl) return

    const controller = new AbortController()
    activeController = controller
    onCleanup(() => controller.abort())
    isLoading.value = true
```

Відповідь належить поточному запуску

useFetch.js · рядки 38–44

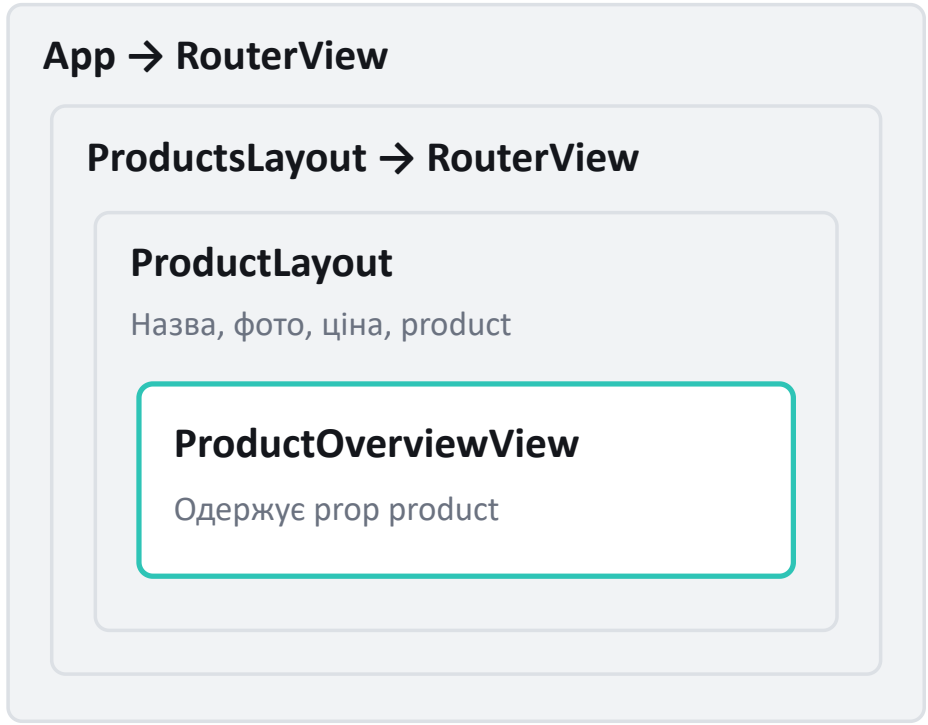
```
      const result = await response.json()
      // Застаріла відповідь не має права змінювати поточний стан.
      if (!controller.signal.aborted && activeController === controller)
        data.value = result
    } catch (cause) {
      if (!controller.signal.aborted && activeController === controller)
        error.value = cause
```

Стани сторінки товару

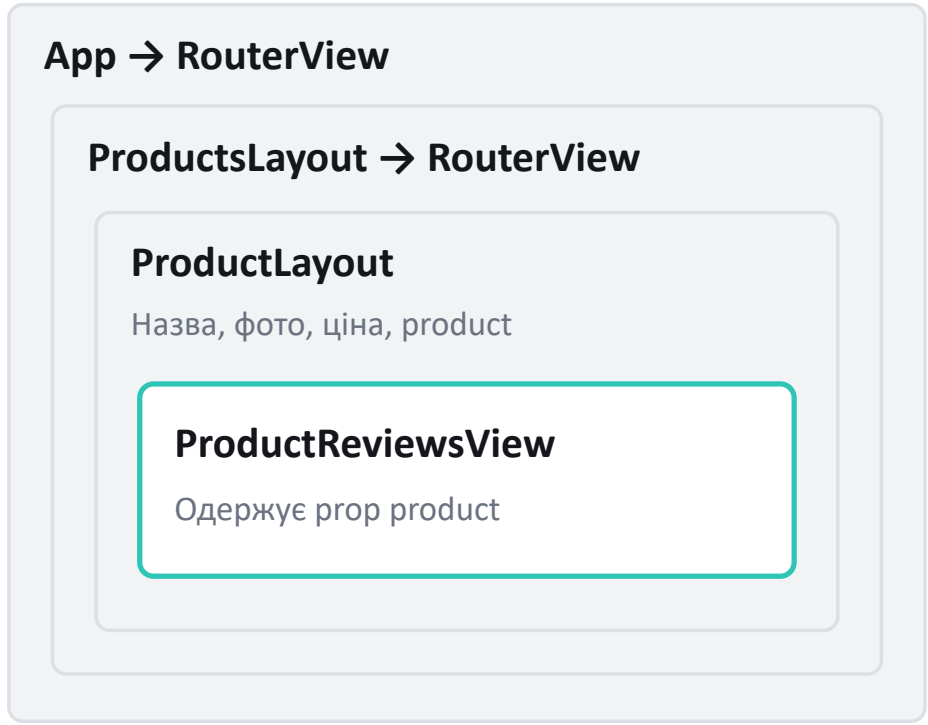
Поняття	Зміст
Завантаження	Поточний id і повідомлення про очікування.
HTTP 404 / некоректний id	Товар не знайдено.
Інша помилка	Повідомлення та повтор через refetch.
Успішний товар	Спільна інформація, огляд або відгуки.
Порожні reviews	Окреме повідомлення всередині розділу відгуків.

Оболонка та вкладені розділи

/products/12



/products/12/reviews



ProductLayout зберігає спільні дані, RouterView перемикає дочірній вміст.

children і порожній дочірній шлях

routes.js · рядки 3–19

```
{
  path: '/products',
  component: () => import('../layouts/ProductsLayout.vue'),
  meta: { breadcrumb: { label: 'Каталог', name: 'products' } },
  children: [
    { path: '', name: 'products', component: () => import('../views/ProductCatalogView.vue') },
    {
      path: ':id',
      component: () => import('../layouts/ProductLayout.vue'),
      meta: { breadcrumb: { dynamic: 'product', name: 'product-overview' } },
      children: [
        { path: '', name: 'product-overview', component: () => import('../views/ProductOverviewView.vue') },
        { path: 'reviews', name: 'product-reviews', component: () => import('../views/ProductReviewsView.vue'),
          meta: { breadcrumb: { label: 'Відгуки', name: 'product-reviews' } } },
      ],
    },
  ],
```

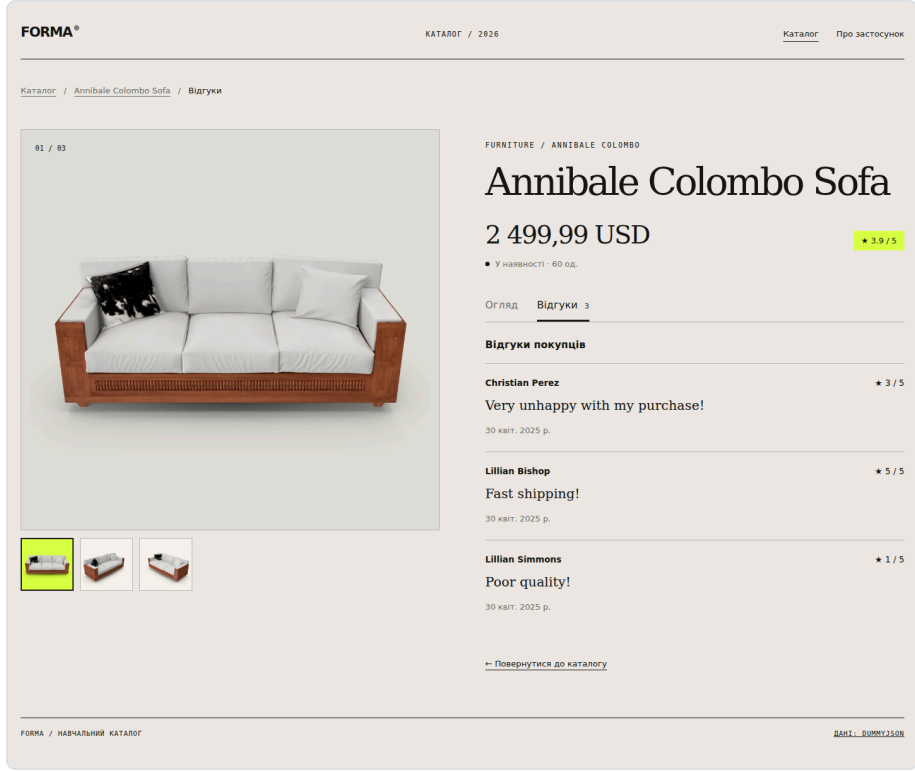
Один product для огляду й відгуків

ProductLayout.vue · рядки 67–71

```
      <nav class="product-sections" aria-label="Розділи товару">
        <RouterLink :to="location('product-overview')">Огляд</RouterLink>
        <RouterLink :to="location('product-reviews')">Відгуки <span>{{ product.reviews?.length || 0 }}</span></RouterLink>
      </nav>
      <RouterView v-slot="{ Component }"><component :is="Component" :product="product" /></RouterView>
```

Хлібні крихти на сторінці відгуків

1. **Каталог** Посилання на список із збереженим пошуком.
2. **Назва товару** Посилання на його огляд.
3. **Відгуки** Текст поточного рівня.



Робочий приклад FORMA з даними DummyJSON.

route.matched і route.meta

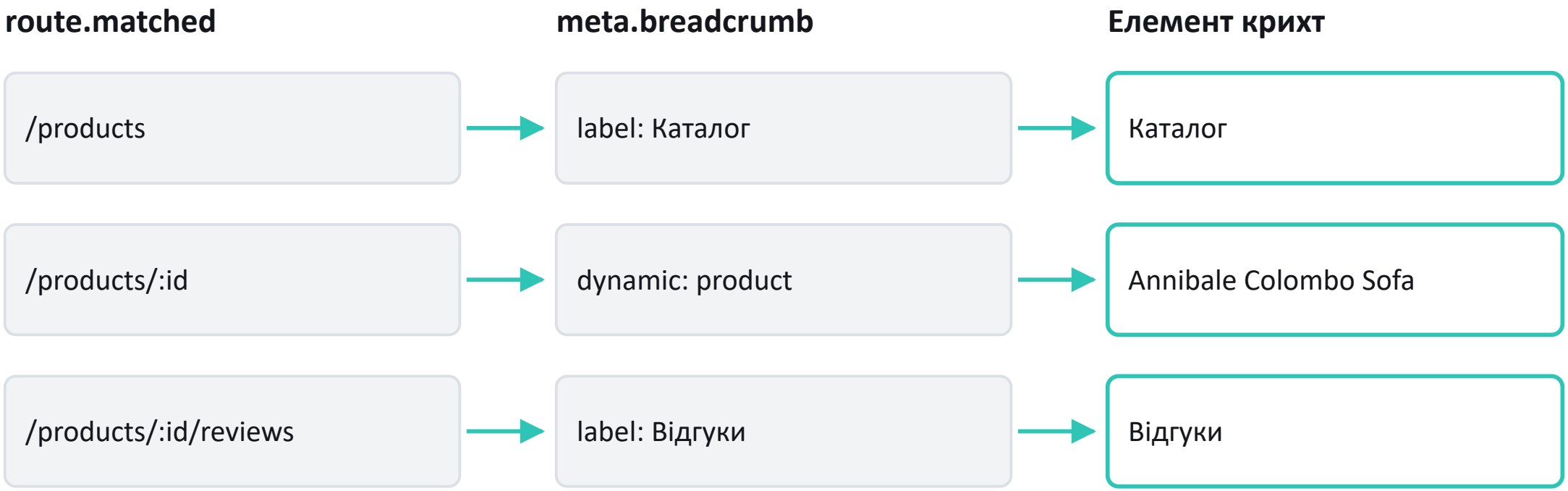
Поняття	Зміст
route.matched	Масив зіставлених записів від предка до нащадка.
record.meta	Власні метадані одного рівня.
route.meta	Поверхневе об’єднання метаданих усіх рівнів.
Однаковий ключ	Дочірнє значення перекриває батьківське в route.meta.

Власний контракт meta.breadcrumb

routes.js · рядки 3–16

```
  {
    path: '/products',
    component: () => import('../layouts/ProductsLayout.vue'),
    meta: { breadcrumb: { label: 'Каталог', name: 'products' } },
    children: [
      { path: '', name: 'products', component: () => import('../views/ProductCatalogView.vue') },
      {
        path: ':id',
        component: () => import('../layouts/ProductLayout.vue'),
        meta: { breadcrumb: { dynamic: 'product', name: 'product-overview' } },
        children: [
          { path: '', name: 'product-overview', component: () => import('../views/ProductOverviewView.vue') },
          { path: 'reviews', name: 'product-reviews', component: () => import('../views/ProductReviewsView.vue'),
            meta: { breadcrumb: { label: 'Відгуки', name: 'product-reviews' } } },
```

Записи маршрутів стають крихтами



Каталог / Annibale Colombo Sofa / Відгуки

Кожен запис надає власний підпис або правило отримання динамічної назви.

Побудова елементів у composable

useBreadcrumbs.js · рядки 5–21

```
export function createBreadcrumbItems(route, productTitle = '') {
  return route.matched.flatMap((record) => {
    const crumb = record.meta.breadcrumb
    if (!crumb) return []
    return [{
      label: crumb.dynamic === 'product'
        ? productTitle || `Товар №${route.params.id}`
        : crumb.label,
      to: {
        name: crumb.name,
        ...(crumb.dynamic === 'product' || crumb.name === 'product-reviews'
          ? { params: { id: route.params.id } } : {}),
        ...(route.query.q ? { query: { q: route.query.q } } : {}),
      },
    }]
  })
}
```

Реактивність useBreadcrumbs

useBreadcrumbs.js · рядки 23–26

```
export function useBreadcrumbs(productTitle = '') {
  const route = useRoute()
  return computed(() => createBreadcrumbItems(route, toValue(productTitle)))
}
```

Реактивна назва товару в крихтах

ProductLayout.vue · рядки 12–14

```
const isMissing = computed(() => !validId.value || error.value?.status === 404)
const productTitle = computed(() => isMissing.value ? 'Товар не знайдено' : product.value?.title || '')
const breadcrumbs = useBreadcrumbs(productTitle)
```

Доступна розмітка Breadcrumbs

Breadcrumbs.vue · рядки 5–13

```vue
<nav class="breadcrumbs" aria-label="Навігаційний ланцюжок">
  <ol class="breadcrumbs__list">
    <li v-for="(item, index) in items" :key="item.to.name" class="breadcrumbs__item">
      <span v-if="index" class="breadcrumbs__separator" aria-hidden="true">/</span>
      <span v-if="index === items.length - 1" class="breadcrumbs__current" aria-current="page">{{ item.label }}</span>
      <RouterLink v-else class="breadcrumbs__link" :to="item.to">{{ item.label }}</RouterLink>
    </li>
  </ol>
</nav>
```

Чернетка вводу та підтверджений пошук

Поняття	Зміст
query	Поточний текст поля; одразу скасовує старий запит.
route.query.q	Підтверджена умова пошуку в URL.
debouncedInput	Знімок вводу після паузи 400 мс.
products / skip	Локальні завантажені сторінки каталогу.

Debounce підтверджує пошук через replace

ProductCatalogView.vue · рядки 32–45

```
watch(query, () => {
  if (restoringQuery) return
  searchDirty.value = true
  abort()
}, { flush: 'sync' })
watch(debouncedInput, async ({ query: value }) => {
  if (!searchDirty.value || value.trim() !== query.value.trim()) return
  if (value.trim() === routeQuery.value) {
    refetch()
  } else {
    await router.replace({ name: 'products', query: { ...route.query, q: value.trim() || undefined } })
  }
  searchDirty.value = false
})
```

Маршрут відновлює поле та список

ProductCatalogView.vue · рядки 51–59

```
watch(routeQuery, (value) => {
  restoringQuery = true
  query.value = value
  restoringQuery = false
  searchDirty.value = false
  skip.value = 0
  products.value = []
  total.value = 0
}, { flush: 'sync' })
```

Повторний пошук A → B → A

1. **Запит A** Операція вже виконується.
2. **Введено B** Сирий ввід скасовує запит A.
3. **Повернення до A** Кінцевий текст знову дорівнює поточному q.
4. **Пауза завершилася** Новий знімок вводу запускає refetch для A.

Інформаційна сторінка та catch-all

routes.js · рядки 21–25

```
  { path: '/about', name: 'about', component: () => import('../views/AboutView.vue'),
    meta: { breadcrumb: { label: 'Про застосунок', name: 'about' } } },
  { path: '/:pathMatch(.*)*', name: 'not-found', component: () => import('../views/NotFoundView.vue'),
    meta: { breadcrumb: { label: 'Сторінку не знайдено', name: 'not-found' } } },
]
```

Перехід усередині SPA та reload

Поняття	Зміст
RouterLink	Маршрутизатор змінює URL і показаний компонент.
Пряме відкриття / reload	Сервер одержує запит документа за вкладеним шляхом.
History fallback	Сервер повертає index.html для маршрутів застосунку.
Catch-all у Vue	Показує інтерфейс 404, але не задає HTTP-статус документа.

Реалізація та перевірка лабораторної 6

1. **Маршрути** Відкрити огляд і відгуки напряму та виконати reload.
2. **Спільні дані** Перемкнути розділи без повторного запиту товару; перевірити зміну id.
3. **Крихти** Перейти до огляду й каталогу за посиланнями та з клавіатури.
4. **Пошук** Відновити q через історію та повторити сценарій A → B → A.
5. **Збірка** Виконати npm run build і перевірити готові адреси через vite preview.

Ваші запитання

Vue Router і сторінка товару